A STUDY IN TWO-DIMENSIONAL GRID GENERATION

by

SENTHAN S.

DEPARTMENT OF MECHANICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

June, 1997

A STUDY IN TWO-DIMENSIONAL GRID GENERATION

A Thesis Submitted

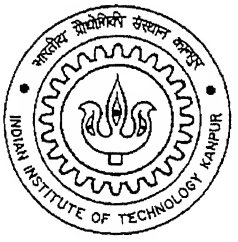
in Partial Fulfillment of the Requirements

for the Degree of

MASTER OF TECHNOLOGY

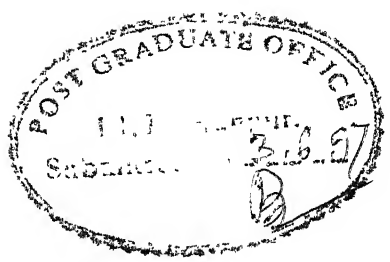
by

SENTHAN S.



DEPARTMENT OF MECHANICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY KANPUR

June, 1997



CERTIFICATE

It is certified that the work contained in the thesis entitled "*A Study in Two-Dimensional Grid Generation*", by *Senthan S.*, has been carried out under our supervision and that this work has not been submitted elsewhere for a degree.

(~~On leave~~)
Dr. Sangeeta Kohli
Assistant Professor
Department of Mechanical Engineering
I.I.T., Kanpur

Dr. V. Eswaran
Associate Professor
Department of Mechanical Engineering
I.I.T., Kanpur

June, 1997.

ACKNOWLEDGEMENT

I express my sincere gratitude to my thesis supervisors **Dr. V. Eswaran** and **Dr. Sangeeta Kohli** for their guidance and support throughout the course work. I owe the greatest debt to them. The time I spent with them is the most fruitful and potential learning period of my life. I am very grateful to their friendly and encouraging nature.

I am deeply indebted to the research scholars in the CFD Lab for their valuable suggestions and help time to time.

Thanks are also due to my classmates and Tamil friends for making my stay at IIT Kanpur a memorable one.

Senthan S.

ABSTRACT

The basic ideas of the construction of numerically generated body-fitted coordinate systems for the finite difference solution of partial differential equations are discussed. Algebraic grid generation and grid generation based on partial differential equations have been discussed in detail with specific reference to Transfinite Interpolation, Laplace, Thomas-Middlecoff, Sorenson, and Modified Sorenson method. Some parameters which affect the 'goodness' of a grid are prescribed. Finally, the grid generated by various methods for various test cases are compared, to evaluate their performance on H-type grids generated for viscous, internal flows. It was found that Thomas-Middlecoff gives the best performance with Transfinite Interpolation next best.

Contents

List of Figures

Nomenclature

A	Cell area
A, B	Arbitrary constants
a, b, c, d	Arbitrary constants
d	Denotes distance
$\imath$	Grid node corresponding to $(\imath - 1)\Delta\xi$
J	Transformation Jacobian matrix
j	Grid node corresponding to $(j - 1)\Delta\eta$
P, Q	Source terms of poisson's equation
r	A vector denotes a point (x, y)
S	Denotes distance
s	Denotes distance
W	Denotes distance
x, y	Cartesian coordinates
$\tilde{\alpha}$, β, γ	Coefficients in transformed poisson's equation
θ	Angle between grid lines (deg)
ξ, η	Curvilinear coordinates
ϕ, ψ	Thomas-Middlecoff grid control functions
$\Delta\xi$	Grid interval in ξ direction
$\Delta\eta$	Grid interval in η direction

Superscripts

n	Represents n^{th} iteration level
$'$	Represents computational domain

Subscripts

b	Represents values at boundary
$\imath$	Represents interior point
max	Represents maximum value
$mın$	Represents minimum value
x	Represents partial derivatives with respect to x keeping y constant
y	Represents partial derivatives with respect to y keeping x constant
ξ	Represents partial derivatives with respect to ξ keeping η constant
η	Represents partial derivatives with respect to η keeping ξ constant

Chapter 1

INTRODUCTION

The generation of body-fitted coordinate systems plays an important role in the accurate computation of numerical solutions of partial differential equations on regions with arbitrary shaped boundaries. A grid which is not well suited for the problem can lead to unsatisfactory results. In some applications, improper choice of grid point locations can lead to instability or lack of convergence, While with a proper selection of grid points, the accuracy of the numerical simulation can be increased without increased computational effort.

1.1 Classification of Grids

Grids can be classified into structured grids and unstructured grids. In structured grid generation, the grid points in the complex physical domain have a one to one correspondence with the grid points of a rectangular grid of the same size in the computational domain. Structured grids are primarily used for obtaining solutions based on finite difference or finite volume methods. The number of nearest neighbours for each interior grid point is fixed in a structured grid (i.e. 2, 4, 6 for one, two and three dimensions respectively).

In unstructured grid generation, no definite structure is maintained between any typical node and its neighbours. Therefore, unstructured grids are best suited for finite element applications.

We are interested only in structured grid generation. So unstructured grid generation will not be discussed here after.

1.2 Importance of Body-fitted Coordinate Systems

With a transformation approach, all computations can be done on a square grid in a transformed domain regardless of the shape of the physical domain. In rectangular enveloped grid systems, the cells adjacent to a complex boundary will be incomplete, as they will approximate the boundary using rectangular cells and interpolation is required over these cells. In body-fitted coordinate systems, the body surface is a boundary in the computational domain which enables better implementation of boundary conditions. Thus body-fitted grid system avoids the need for modifying the solver for a new geometry.

1.3 Present Work

Many packages are available for structured grid generation. Each of them will be based on one or two grid generation methods and are suitable only for certain kinds of problems. So it is difficult to get a package for universal application. These packages are generally a black box, and we do not exactly know what is going inside. Also, for a research group, it is desirable to have a package of our own to suit our needs which can be fine tuned as and when desired.

So, in our present work, we developed a code in FORTRAN 77 for structured grid generation in two dimensions using various methods. The code can also evaluate the 'goodness' of the grid using various criteria which are discussed in chapter 2. Finally, the grids generated by various techniques are compared in chapter 4.

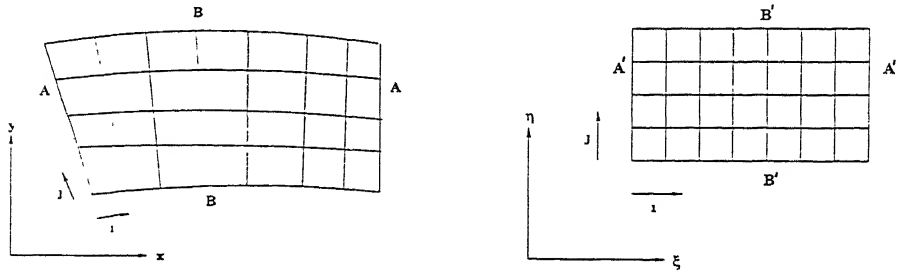
Chapter 2

PRINCIPLES OF GRID GENERATION

The fundamental problem in grid generation is to systematically choose a grid which allows for an accurate numerical solution of the governing equations, with the minimum computation. The physical domain on which the problem is to be solved, is often mapped onto a rectangular computational domain. Then the problem is solved in the computational domain by solving the governing equations, suitably transformed, in the rectangular grid. In multiblock grid generation, separate sections of the physical domain are mapped onto separate squares or rectangles.

Figure 2.1 shows a structured grid in the physical and transformed domain. Note that for each grid point (i, j) in the physical domain there is a corresponding grid point in the transformed domain. Also for each grid line ($A - A$ or $B - B$) there are corresponding lines ($A' - A'$, $B' - B'$) in the transformed domain. Further, the grid lines in the transformed domain are lines of constant ξ or lines of constant η.

Various aspects of structured grid generation is discussed in this chapter.



(a) Physical domain

(b) Computational domain

Figure 2.1:

2.1 Requirements of a Computational Grid

For better results of numerical simulations, there are some disirable features required of the grid point distribution. Some of them are.

1. The mapping from physical domain to computational domain must be one-to-one

2. The grid lines should be smooth so as to provide continuous transformation parameters (which appear in the transformed governing equations).

3 Grid points should be closely spaced in the physical domain where large numerical errors are expected. (eg. fine grids are necessary in the boundary layer region).

4. Excessive skewness should be avoided. If the angles between grid lines are close to zero, the numerical simulation will be highly inaccurate due to loss of independence between the grid lines.

5 It is desirable to make the intersecting grid lines orthogonal in the boundary for easy implementation of boundary conditions.

It is not possible to meet all the demands simultaneously . A particular property may be very important, and the requirements may change from problem to problem with some feasible compromise on the others The chief concern in body-fitted grid generation is to ensure smoothness while efficient grid spacing is the primary interest for adaptive grid generation. Apart from the above requirements, grid generation process should be automatic so as to avoid the drudgery of giving grid data manually.

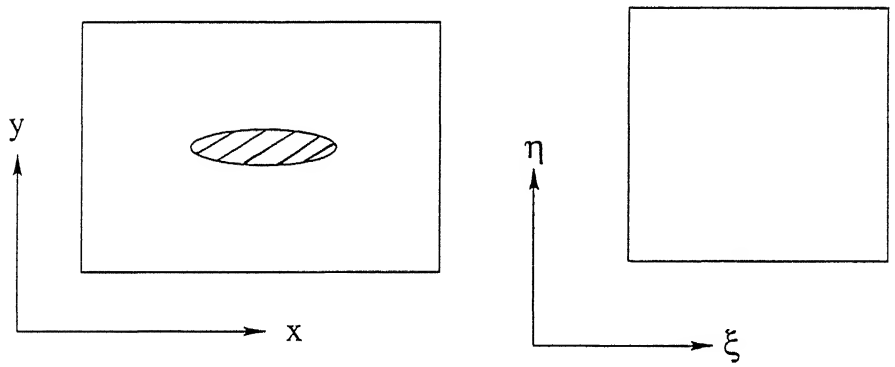
2.2 Classification of Structured Grids

Two dimensional grids which map the physical domain to computational domain are generally classified into three categories. They are:

(a) O-type grids

(b) C-type grids

(c) H-type grids

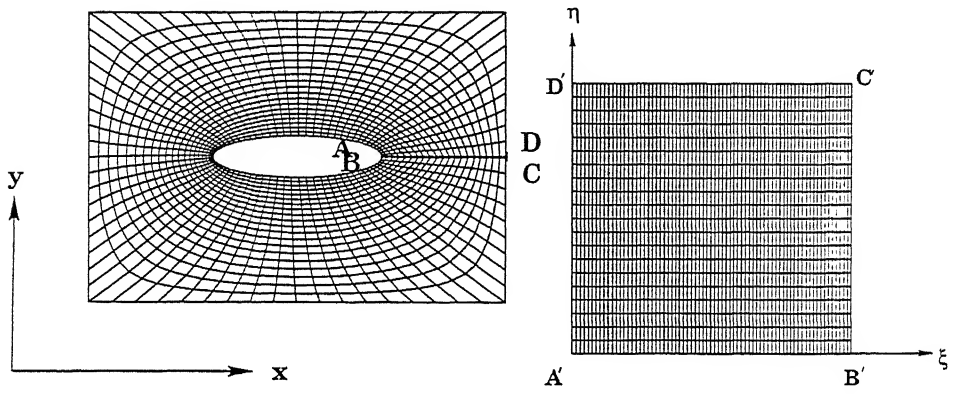


(a) Physical domain

(b) Computational domain

Figure 2.2:

For a given geometry any one of these configurations can be obtained by suitable mapping Consider the domain shown in figure 2 2(a), which contains a 'hole' (eg a solid body) which is to be excluded from the computational domain shown on the right

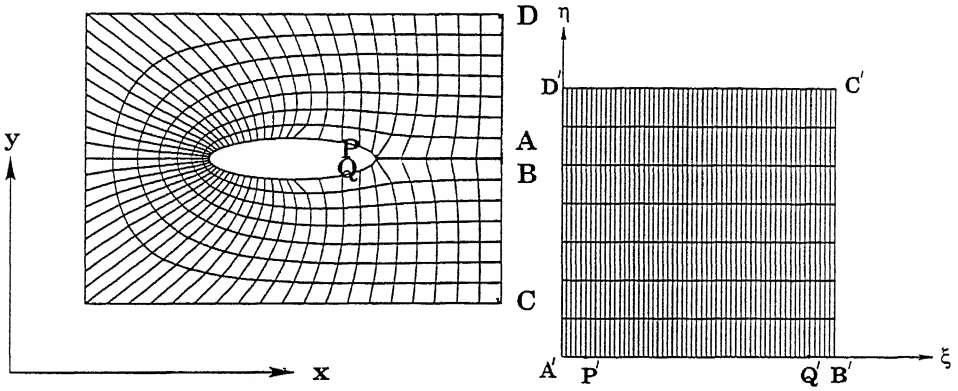


(a) Grids in physical domain (b) computational domain

Figure 2.3: O - Type Grid

In O-type grid generation, the exterior of the physical domain is mapped onto one side of the computational domain. To get O-grids for this domain, shown in figure 2.2(a), a branch cut is introduced in the domain and the points A,B,C,D are chosen in either sides of the branch cut as shown in the figure 2.3(a). Then AB. BC, CD, DA are mapped onto $A'B'$, $B'C'$, $C'D'$, $D'A'$ respectively. Thus the two sides of the branch cut (AD, BC) are mapped onto two opposite sides of the computational domain, and the line $A'B'$ and $B'C'$ physically correspond to the same line (branch cut) It can be seen that in this case, η-constant lines forms O-type configuration in the physical domain.

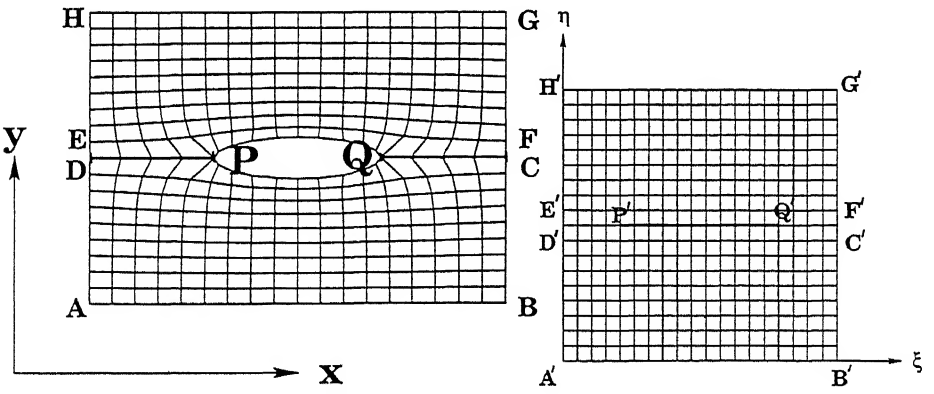
In C-type grid generation, the outer boundary is mapped onto three sides of the computational domain. To generate C-type grids for the same domain shown in figure 2.2(a), a branch cut is introduced and the points A, B are chosen on either side where the branch cut meets the outer boundary. The



(a) Grids in physical domain (b) computational domain

Figure 2.4. C - Type Grid

points C, D are suitably selected in the outer boundary and then AB, BC, CD, DA are mapped onto $A'B', B'C', C'D', D'A'$ respectively. The forward sweep branch cut (AP), object surface (PQ) and reverse sweep branch cut (QB) form one side of transformed region. The solid body is reduced to the line $P'Q'$ at the bottom boundary of the transformed domain. It can be seen that in this case, η-constant lines form a 'C' figure in the physical domain.



(a) Grids in physical domain (b) computational domain

Figure 2.5: H - Type Grid

For H-type grid generation, the outer boundary is mapped onto four sides of the computational domain. To generate H-grids for the same domain shown in figure 2.2(a), branch cuts are introduced on either sides of the object (figure 2.5(a)) and the upper and lower portions ABCQPD and EPQFGH are separately mapped onto $A'B'C'Q'P'D'$ and $E'P'Q'F'G'H'$. The object in the physical domain reduces to a line in the computational domain.

C-grids and O-grids are mostly used for external flow calculations such as flow over airfoils, cascades etc. . while H-grids are mostly used for internal flow calculations.

2.3 Transformation Relations

To construct numerical solution of partial differential equations on regions of arbitrary shape, the computations have to be carried out in the transformed region with the transformed coordinates $(\xi,\ \eta)$ as the independent variables. The coordinate in the physical domain will be denoted by $(x,\ y)$ and those of the transformed region by $(\xi,\ \eta)$. Any point on the physical domain has a single corresponding point on the transformed region, and vice-versa. In this section, the transformation relations between cartesian coordinate system and body-fitted curvilinear coordinate $(\xi,\ \eta)$ system are discussed.

The body-fitting coordinate transformation will be

$$\xi = \xi(x, y)\ ,$$
$$\eta = \eta(x, y)\ ,$$

the bounds of the computational domain are defined by

$$0 \leq \xi \leq 1\ ,$$
$$0 \leq \eta \leq 1\ .$$

From the chain rule

$$\begin{Bmatrix} d\xi \\ d\eta \end{Bmatrix} = \begin{bmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{bmatrix} \begin{Bmatrix} dx \\ dy \end{Bmatrix} \tag{2.1}$$

where ξ_x, ξ_y and η_x, η_y are the partial derivatives of ξ, η with respect to x, y. Also we have the inverse relation ship

$$x = x(\xi, \eta) ,$$
$$y = y(\xi, \eta) .$$

Again using chain rule

$$\begin{Bmatrix} dx \\ dy \end{Bmatrix} = \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix} \begin{Bmatrix} d\xi \\ d\eta \end{Bmatrix} \tag{2.2}$$

substituting Equation (2.1) in Equation (2.2)

$$\begin{Bmatrix} dx \\ dy \end{Bmatrix} = \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix} \begin{bmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{bmatrix} \begin{Bmatrix} dx \\ dy \end{Bmatrix} , \tag{2.3}$$

we see that the two matrices on the right must be inverses of each other, i.e.,

$$\begin{bmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{bmatrix} = \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix}^{-1} = \frac{1}{\mid J \mid} \begin{bmatrix} y_\eta & -x_\eta \\ -y_\xi & x_\xi \end{bmatrix} \tag{2.4}$$

where $\mid J \mid = x_\xi y_\xi - x_\eta y_\eta$ is the determinant of the jacobian matrix

$$[J] = \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix}$$

For a one-to-one mapping, the determinant $\mid J \mid$ should be finite and non-zero. It can be shown that the cell area in the physical domain

$$dA = \mid J \mid d\xi d\eta$$

while the corresponding area in the transformed region is $d\xi d\eta$. Thus the determinant $\mid J \mid$ represents the local area scaling factor.

From the Equation 2.4, The relations between the partial derivatives of curvilinear coordinates (ξ, η) and that of cartesian coordinates (x, y) are obtained as

$$\xi_x = \frac{y_\eta}{|J|}, \tag{2.5}$$

$$\xi_y = \frac{-x_\eta}{|J|}, \tag{2.6}$$

$$\eta_x = \frac{-y_\xi}{|J|}, \tag{2.7}$$

and

$$\eta_y = \frac{x_\xi}{|J|}. \tag{2.8}$$

These relations are very useful in transforming the governing equations from the (x, y) coordinate system to (ξ, η) system.

The governing equations will be in terms of the physical domain variables (x, y). However, as they will be solved in the transformed domain, these equations need to be put in terms of (ξ, η). This is done by the mapping

$$x = x(\xi, \eta)$$
$$y = y(\xi, \eta)$$

and

$$\left(\frac{\partial \phi}{\partial x}\right)_\eta = \left(\frac{\partial \phi}{\partial \xi}\right)_\eta \left(\frac{\partial \xi}{\partial x}\right)_y + \left(\frac{\partial \phi}{\partial \eta}\right)_\xi \left(\frac{\partial \eta}{\partial y}\right)_x$$
$$= \left(\frac{\partial \phi}{\partial \xi}\right)_\eta \frac{y_\eta}{|J|} - \left(\frac{\partial \phi}{\partial \eta}\right)_\xi \frac{y_\xi}{|J|} \qquad \text{etc.}$$

so that the equations are finally expressed only in terms of (ξ, η) derivatives with respect to (ξ, η). The transformed equations are normally much more complicated than the original equations. This price has to be paid for the simplification in geometry.

2.4 Grid Generation Techniques

The techniques available for structured grid generation can be divided into three major categories. They are

1. Algebraic grid generation
2. Conformal mapping
3. Methods using Partial Differential Equations

2.4.1 Algebraic grid generation

Among the grid generation techniques, algebraic grid generation computationally most efficient. These methods are based on mathematical interpolation functions. Explicit control of grid shape and grid spacing is also possible. The disadvantage of these methods is that the slope discontinuities present at the boundaries tend to propagate inside and hence smoothness is not assured.

Some of the well known algebraic grid generation methods are transfinite interpolation, multisurface method and two boundary method. Transfinite interpolation is most widely used method. This method is discussed in detail in the next section.

2.4.2 Conformal mapping

Conformal mapping provides a smooth orthogonal grid except at a few singular points. This method can be used only for two dimensional problems. In this thesis, we are not concerned with this method and hence it is not discussed here.

2.4.3 Grid generation based on Partial Differential Equations

These methods are the most commonly used. These methods can be further classified into elliptic, hyperbolic and parabolic grid generation depending upon the generation equations used. The grids generated using this methods are smooth, and control of orthogonality and grid spacing is possible. But these methods are computationally expensive.

Elliptic grid generation uses as grid generation equations Laplace or Poisson's equations, where all four boundaries need to be specified. Hence this method is used for closed geometries. Elliptic grid generation will be discussed in the next chapter in detail.

In hyperbolic grid generation, the outer boundary need not be specified. The grid lines generated using hyperbolic equations are orthogonal near the boundary. These methods are used to generate grids for domains that are open.

Parabolic grid generation is not frequently used. Achieving orthogonality is not effective as hyperbolic method. Also, control functions are not developed for this method. Both parabolic and hyperbolic grid generation are faster than elliptic grid generation.

2.5 Transfinite Interpolation

This is a powerful but simple algebraic technique for grid generation and is frequently used. The interior coordinates are generated by series of unidirectional interpolation between prescribed boundary data.

Consider a 2-D physical domain ABCD shown in figure 2.6(a). This is

to be mapped onto a square computational domain with equally spaced grid points as shown in figure 2.6(e)

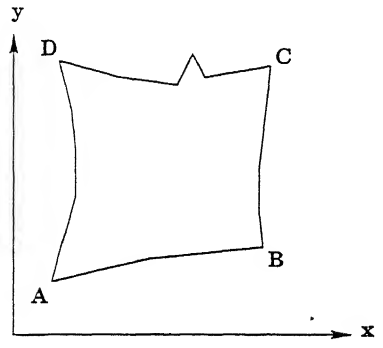
First, grid points are placed on the boundary in the physical domain. This is done by choosing the grid points on the boundaries according to some sensible criterion Fig 2.6(b). The number of grid points on opposite sides of the domain should be equal. These boundary grid points have a one-to-one correspondence with the boundary points on the transformed domain Fig 2.6(e). The (ξ, η) values of each grid point in Fig 2.6(e) are known as points are equally spaced and the domain is a unit square in (ξ, η) space. Now we need to find the (x, y)coordinates for every interior grid point in Fig 2.6(b), corresponding to each grid point in Fig 2.6(e).

Now unidirectional interpolation is applied in any one of the direction, say the ξ-direction. For this we do the following. For every $\eta = constant$ line in figure 2.6(e) interpolate the (x, y) coordinates of a point P using

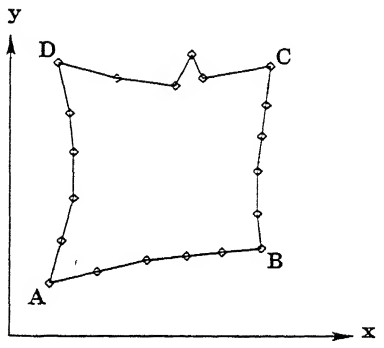
$$x'_P = (1 - \xi)x_E + \xi x_F$$

$$y'_P = (1 - \xi)y_E + \xi y_F$$

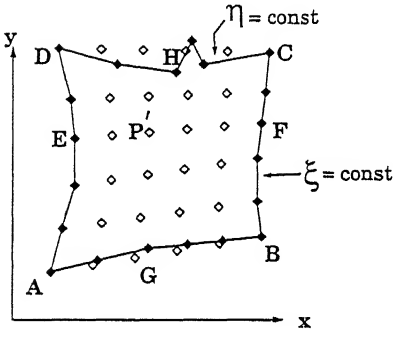
Where E and F are the end of the $\eta - constant$ line. This interpolation is carried out for all $\eta - constant$ lines including the boundary lines $\eta = 0$ and $\eta = 1$. These interpolated points are shown in the figure 2.6(c). It can be seen in this figure that, while the boundary points at the $\xi = 0$ and $\xi = 1$



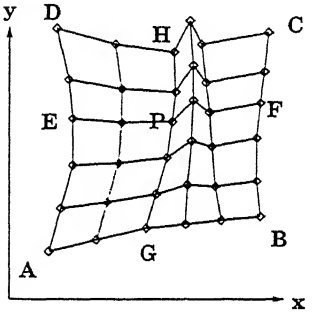
(a) Physical domain



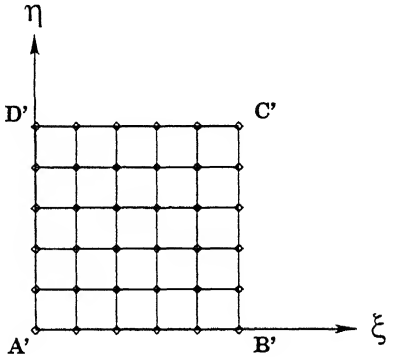
(b) Chosen boundary points



(c) After unidirectional interpolation in ξ - direction



(d) Final grid



(e) Computational domain

Figure 2.6. Transfinite Interpolation

boundaries (AD and BC) the interpolated boundary points match exactly with the chosen boundary points, at the boundaries AB and CD, the grid points obtained by unidirectional interpolation need not coincide with the chosen boundary points and may even fall outside the domain.

This anomaly is corrected by unidirectional interpolation in the η direction. These interpolation are made along each ξ – *constant* line. The corrections Δx_P and Δy_P for each point P are made such that at the $\eta = 0$ and $\eta = 1$ boundaries the corrections are exactly the difference $(\Delta x_G,\ \Delta y_G)$ and $(\Delta x_H,\ \Delta y_H)$ which would bring the 'boundary' points obtained by the first interpolation exactly to the chosen boundary points.

$$\Delta x_P = (1-\eta)\Delta x_G + \eta \Delta x_H$$

$$\Delta y_P = (1-\eta)\Delta y_G + \eta \Delta y_H$$

Where G and H are the end points of the $\xi = constant$ line on which P lies. Δx_G, Δx_H, Δy_G and Δy_H are the corrections for the boundary points G and H and is given by

$$\Delta x_G = x'_G - x_G \qquad \Delta y_G = y'_G - x_G$$
$$\Delta x_H = x'_H - x_H \qquad \Delta y_H = y'_H - x_H$$

Where $x'_G - x_G$ etc. are the values obtained by the first interpolation and x_G etc. are the values chosen for the boundary points.

The final values of the coordinates of P are obtained as

$$x_P = x'_P - \Delta x_P$$

$$y_P = y'_P - \Delta y_P$$

The final mesh in the physical domain will appear as shown in the figure 2.6(d), each point on which has a corresponding point in the transformed domain. Now as the $(x_P,\ y_P)$ values in the physical domain, and the $(\xi_P,\ \eta_P)$ values

in the transformed domain, are known for every point P, the relationship $x = x(\xi, \eta)$ and $y = y(\xi, \eta)$ are also known.

The quality of the mesh generated entirely depends upon the chosen boundary points distribution. So care should be taken in the placement of the boundary points. Transfinite interpolation can be applied to 2-D and 3-D geometries and for all grid configurations ie O - type, C - type and H type grids.

2.6 Evaluation of Grid Quality

Truncation error of numerical solutions depends upon the grid upon which the discretized approximation of the differential equation is solved. A desirable grid should give minimum truncation error. Certain properties of the grid are found to have a strong effects on the truncation error [5]. Parameters which affect these properties can be used to express the goodness of a grid. The parameters are

1. Transformation jacobian
2. Skewness
3. Aspect ratio
4. Adjacent cell ratio
5. Nearness parameter

Transformation Jacobian

As we have already mentioned, the transformation Jacobian represents the local area scaling factor. So the Jacobian should not become zero for the

mapping to be one-to-one.

Skewness

Skewness is a measure of nonorthogonality between two intersecting grid lines. Higher the skewness, higher will be the truncation error due to loss of independence between the grid lines. Many methods are available to evaluate the cell skewness. They are

(a) Intersection Angle

The intersection angle θ between two grid lines, as shown in figure 2.7, is calculated. It is desirable that the intersection angle be 90°, but this is not always possible. At least it is desirable to maintain the intersection angle in the range

$$45^{\circ} \leq \theta \leq 135^{\circ}$$

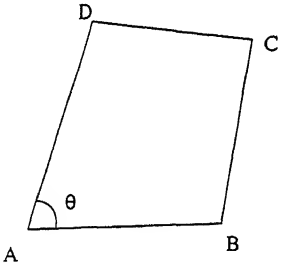


Figure 2.7:

(b) Based on length of diagonals

In this method skewness is calculated by using the length of the diagonals of a grid cell. For a grid cell ABCD as shown in figure 2.8 skewness is calculated

as

$$skewness = 1 - \frac{length\ of\ smaller\ diagonal}{length\ of\ larger\ diagonal} \frac{BD}{AC}$$

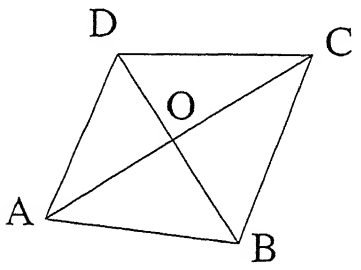


Figure 2.8:

(c) Based on length of half diagonal

For a grid cell of shape as shown in figure 2.9, the previous method of calculating cell skewness will fail to give true information about the cell skewness for the diagonals are equal and the skewness will be zero, although it can be seen the quadrilateral is quite skewed. For such cases, the length of the half diagonals is used to calculate the skewness.

$$skewness = 1 - \frac{length\ of\ smallest\ half\ diagonal}{length\ of\ larger\ half\ diagonal\ of\ the\ opposite\ diagonal} \frac{OD}{OA}$$

(d) Based on area of triangles

In this method, skewness is calculated using the area of the triangles formed by the diagonals and the four sides of a grid cell.

For a grid shown in figure 2 9

$$skewness = \frac{area\ of\ the\ smallest\ triangle}{area\ of\ biggest\ triangle} \frac{\triangle ODC}{\triangle OAB}$$

In the above three methods, normally the maximum allowable limit for skewness is taken around 0.5

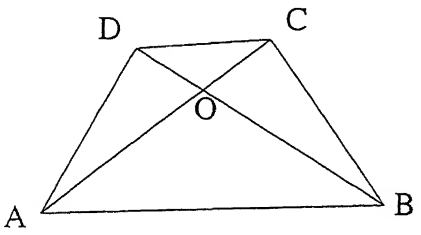


Figure 2.9:

Aspect ratio

This is the maximum ratio between of any two adjacent sides of a cell. Aspect ratio of less than 6 is desirable. It should not exceed 20.

Adjacent cell ratio

This is a measure of grid smoothness. It is a ratio of the areas of two adjacent cells. It is desirable to have it close to one.

Nearness Parameter

This is a measure of how far the first grid lines are from the boundaries or how well does the grid intervals chosen at the boundary penetrate the interior.

Fig 2.10 shows calculation of nearness parameter for a cell on the $\eta = \eta_{max}$ boundary. Let the distance between this boundary and its adjacent grid line be S_i at the cell and S_b at the corner. Also let the length of the grid line AB be W_i at the cell, and the length of the grid line at the edge CD be W_b. Then the nearness parameter is calculated as

$$NearnessParameter = \frac{S_i/W_i}{S_b/W_b}$$

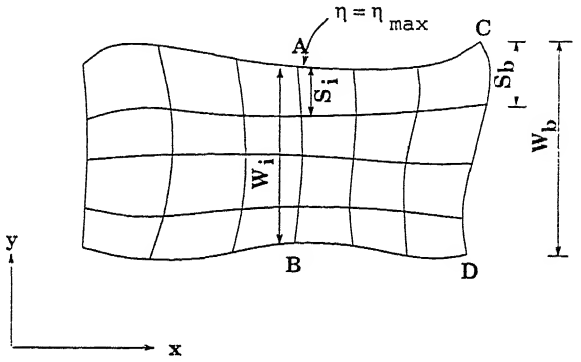


Figure 2.10: Nearness parameter

Among the two $\xi = constant$ boundaries, $\xi = \xi_{min}$ and $\xi = \xi_{max}$, the one which has the maximum S_b is chosen for this calculation

Chapter 3

ELLIPTIC GRID GENERATION SYSTEMS

In mid 1970s Thompson et al [15] first used elliptic Partial Differential Equations for numerical generation of body-fitted coordinates. A number of grid generation packages were developed using Laplace or Poisson's equations. Different functions for controling grid point distribution in the interior were tried. These functions are essentially the source terms of the Poisson's equations. Much effort was made by various authors to calculate the control functions automatically for achieving desired properties of the grid.

In this chapter, elliptic grid generation systems and various control functions developed by various authors are discussed.

3.1 Grid Generation Equations

Elliptic grid generation is based on Poisson's equations of the form

$$\nabla^2 \xi = P \tag{3.1}$$

$$\nabla^2 \eta = Q \tag{3.2}$$

or Laplace equations of the form

$$\nabla^2 \xi = 0 \tag{3.3}$$

$$\nabla^2 \eta = 0 \tag{3.4}$$

where

$$\nabla^2 \xi \equiv \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$

By solving the above equations, using either Dirichlet conditions (grid points specified) or Neumann conditions (spacing specified) at the boundaries, we obtain as solution of the mapping

$$\xi = \xi(x, y) , \tag{3.5}$$

and

$$\eta = \eta(x, y) . \tag{3.6}$$

Once this is obtained we can find the inverse mapping

$$x = x(\xi, \eta) , \tag{3.7}$$

$$y = y(\xi, \eta) . \tag{3.8}$$

and find the (x_P, y_P) correspondence to each grid point P in the square transformed domain where (ξ_P, η_P) are integer multiples of $\Delta\xi$ and $\Delta\eta$, respectively. $(\Delta\xi = 1/M , \Delta\eta = 1/N)$ where M, N are the number of grid intervals in the ξ and η directions respectively. Once we know (x_P, y_P) for each grid points, we have generated the grid and solved our problem. However , an easier way to do this is to convert Equations 3.1-3.4 so that x, y are the dependent variables and ξ, η are the independent variables, as the values of the latter at the grid points are already known. On solving this transformed equations, we obtain Equations 3.7 and 3.8 directly, and thus know the grid in the physical domain.

So these equations are then transformed to the computational domain using the transformation relations discussed in the section 2.3, and the Equations 3.1 and 3.2 transform to

$$A(\alpha x_{\xi\xi} - 2\beta x_{\xi\eta} + \gamma x_{\eta\eta}) + B(\alpha y_{\xi\xi} - 2\beta y_{\xi\eta} + \gamma y_{\eta\eta}) = P \tag{3.9}$$

$$C(\alpha x_{\xi\xi} - 2\beta x_{\xi\eta} + \gamma x_{\eta\eta}) + D(\alpha y_{\xi\xi} - 2\beta y_{\xi\eta} + \gamma y_{\eta\eta}) = Q \tag{3.10}$$

where

$$A = \frac{-y_\eta}{|J|^3}, \qquad B = \frac{x_\eta}{|J|^3}$$

$$C = \frac{y_\xi}{|J|^3}, \qquad D = \frac{-x_\xi}{|J|^3}$$

$$\alpha = x_\eta^2 + y_\eta^2$$

$$\beta = x_\xi x_\eta + y_\eta y_\xi$$

$$\gamma = x_\xi^2 + y_\xi^2$$

and

$$|J| = (x_\xi y_\eta - x_\eta y_\xi)$$

Further simplification yields a system of two elliptic equations of the form

$$\alpha x_{\xi\xi} - 2\beta x_{\xi\eta} + \gamma x_{\eta\eta} = -|J|^2 (P x_\xi + Q x_\eta) \tag{3.11}$$

$$\alpha y_{\xi\xi} - 2\beta y_{\xi\eta} + \gamma y_{\eta\eta} = -|J|^2 (P y_\xi + Q y_\eta) \tag{3.12}$$

These equations are solved by finite difference methods on the uniform grid in $(\xi,\ \eta)$ space. These equations are non-linear since the coefficients α, β, γ are also functions of unknown variables $x(\xi,\ \eta)$, $y(\xi,\ \eta)$ and hence these equations need to be solved iteratively. Gauss Seidel iteration with under relaxation or successive line under relaxation can be used. We have used Gauss Seidel under relaxation in our program.

In our computer program, all the second derivatives are centrally differenced. The coefficients α, β, γ are also calculated using central-difference

formulas using previous iteration values. To satisfy diagonal dominance of the discretised scheme, the first derivatives are differenced using upwind scheme i.e. if P is +ve then x_ξ, y_ξ are discretised using forward difference, else backward difference formula is used, and similarly x_η, y_η are discretised depending upon the values Q If the first derivatives are also approximated using central difference, the scheme will become unstable.

The boundary conditions are given by initially choosing the grid points at the boundaries (Dirichlet conditions). Grid spacing and grid line slope can also be specified as the (Neumann) boundary conditions. But for internal flow applications, since all boundaries are fixed, only Dirichlet boundary conditions can be used.

3.2 Properties of Elliptic Grid Generation Systems

3.2.1 Laplace Systems

An important property of a Laplacian generated grid is its inherent smoothness. Due to this excellent smoothing property, boundary slope discontinuities, if any, are not propagated into the field. The Laplacian system also guarantees that the mapping obtained as the solution to the Elliptic equations is one-to-one. This property is very important to avoid ambiguity in the placement of grid points.

With the Laplace system, grid lines will tend to be equally spaced in the absence of boundary curvature but it will become more closely spaced over convex boundaries and less closely spaced over concave boundaries. This lack of control over grid spacing may be problematic – often finer grid intervals are required in certain regions, eg. near solid boundaries etc., which laplace system

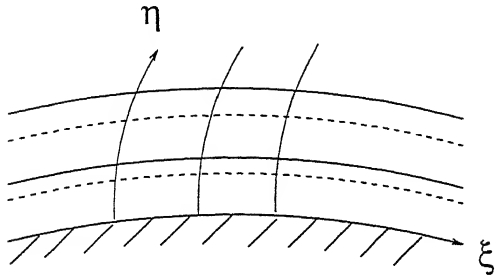


Figure 3.1. Effect of negative Q

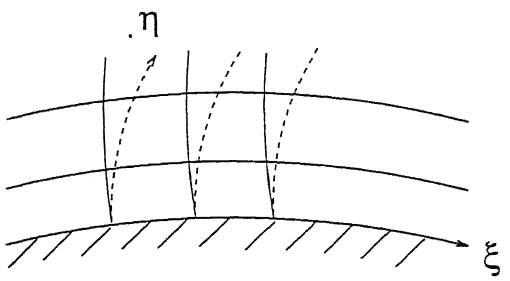


Figure 3.2: Effect of negative P

cannot deliver. This is the primary disadvantage that Laplace sysytems have in comparison with Poisson systems.

3.2.2 Poisson Systems

The source terms P and Q in the Poisson's system can control the spacing between grid lines and the curvature of the grid lines at the boundaries. If we use Poisson's equations with -ve values of Q near the boundary for the figure 3.1, it will make η_{yy} more negative and hence $\eta - constant$ lines are further closely spaced near the boundary. Positive values of Q will have the opposite effort.

Similarly the source term P controls the curvature (and hence the spacing) between $\xi - constant$ lines. Since the boundary points are fixed, the $\xi - constant$ lines can not change its intersection points in the boundary. The overall effort of P is to rotate the $\xi - constant$ lines in the direction of increasing ξ at $\eta - constant$ boundary.

3.3 Control Functions

Many methods are available to determine the control functions P, Q, in Poisson systems. In this section some of them, which are used in our code, are discussed.

3.3.1 Attraction of Grid Lines to Grid Lines/Points

In the TOMCAT code of Thompson et al [12] they used source terms to attract the grid lines towards other grid lines/points.

To achieve small spacing between $\xi - constant$ lines in the neighbourhood of $\xi = \xi_i$, they used the control functions of the form

$$P(\xi,\ \eta) = -A\ sign(\xi - \xi_i)\ \exp\{-B \mid \xi - \xi_i \mid\} \tag{3.13}$$

The constant A is equal to the maximum magnitude of P and it controls the extent by which the $\xi - constant$ lines shift towards $\xi = \xi_i$ while the range of the attraction effect is determined by the decay factor B. If a -ve value of A is used, instead of attraction, the grid lines will be repelled.

To attract the $\xi - constant$ lines towards a given point $(\xi_i,\ \eta_i)$, the control function P takes the form

$$P(\xi,\ \eta) = -A\ sign(\xi - \xi_i)\ \exp\left\{-B\left[(\xi - \xi_i)^2 + (\eta - \eta_i)^2\right]^{\frac{1}{2}}\right\} \tag{3.14}$$

Similarly the functions

$$Q(\xi,\ \eta) = -A\ sign(\eta - \eta_i)\ \exp\{-B \mid \eta - \eta_i \mid\} \tag{3.15}$$

is used to attract $\eta - constant$ lines towards $\eta = \eta_i$ line and the function

$$Q(\xi,\ \eta) = -A\ sign(\eta - \eta_i)\ \exp\left\{-B\left[(\xi - \xi_i)^2 + (\eta - \eta_i)^2\right]^{\frac{1}{2}}\right\} \tag{3.16}$$

is used to attract η – *constant* lines towards the given point $(\xi_i,\ \eta_i)$.

It should be noted that the constant A should not be too large, If it is, the mapping may no longer be one-to-one and grid lines of the same family (eg constant ξ lines) may cross each other.

3.3.2 Thomas - Middlecoff Method

Due to the strong smoothing effect of the Laplacian operator the grid lines in the interior will tend to be equally spaced irrespective of the boundary point distribution. This is not always desirable. In 1980, Thomas and Middlecoff [11] proposed a direct control method whereby the resulting interior grid point distribution is controlled entirely by a prior selection of the grid point distribution along all the boundaries. The angle of intersection between the transverse grid lines and the boundaries can also be controlled.

Thomas and Middlecoff chose source terms of the form

$$P = \phi(\xi,\ \eta)\ (\xi_x^2 + \xi_y^2)$$
$$Q = \psi(\xi,\ \eta)\ (\eta_x^2 + \eta_y^2)$$

where the functions ϕ, ψ will be calculated from boundary point distribution. Introducing these terms, the Poisson's Equations 3.11 and 3.12 assume the form

$$\alpha(x_{\xi\xi} + \phi x_\xi) - 2\beta x_{\xi\eta} + \gamma(x_{\eta\eta} + \psi x_\eta) = 0 \qquad (3.17)$$
$$\alpha(y_{\xi\xi} + \phi y_\xi) - 2\beta y_{\xi\eta} + \gamma(y_{\eta\eta} + \psi y_\eta) = 0 \qquad (3.18)$$

The functions ϕ and ψ are obtained by imposing two constraints i.e. the grid lines transverse to the boundary are (a) locally straight and (b) orthogonal to the boundary.

To find an expression for ϕ along $\eta = \eta_b = constant$ boundaries, ψ is eliminated from the Equations 3.17 and 3.18. This yields

$$\alpha\left[y_\eta\left(x_{\xi\xi}+\phi x_\xi\right)-x_\eta\left(y_{\xi\xi}+\phi y_\xi\right)\right]=y_\eta^2\left[2\beta\left(\frac{x_\eta}{y_\eta}\right)_\xi+\gamma\left(\frac{x_\eta}{y_\eta}\right)_\eta\right] \tag{3.19}$$

The ratio $\frac{x_\eta}{y_\eta}$ represents the slope of the $\xi - constant$ lines that are transverse to the boundary $\eta = \eta_b$. The first constraints is that the transverse coordinate lines is straight (i.e. have zero curvature) in the neighbourhood of the boundary. Mathematically

$$\frac{\partial}{\partial\eta}\left(\frac{x_\eta}{y_\eta}\right)=0 \; at \; \eta \; = \; \eta_b$$

The second condition is that the transverse grid lines are locally orthogonal to the boundary $\eta = \eta_b$. Let a vector $r = (x,\ y)$ denote a point $(x,\ y)$ in physical domain. Then the vector that is locally tangent to $\eta - constant$ line is

$$r_\xi = (x_\xi,\ y_\xi)\ .$$

Similarly the local tangent vector to $\xi - constant$ line is

$$r_\eta = (x_\eta,\ y_\eta)\ .$$

The two families of coordinate lines are orthogonal if and only if

$$r_\xi . r_\eta = 0$$

or

$$x_\xi x_\eta + y_\xi y_\eta = 0 \tag{3.20}$$

Under this orthogonality condition $\beta = 0$. Hence both the terms in the right hand side of the Equation 3.19 vanish at the boundary $\eta = \eta_b$. The Equation 3.20 is used to eliminate the first derivatives $x_\eta,\ y_\eta$ from the Equation 3.19. This yields an expression for ϕ as

$$\phi=\frac{-\left(x_\xi x_{\xi\xi}+y_\xi y_{\xi\xi}\right)}{\left(x_\xi^2+y_\eta^2\right)} \qquad \text{at} \qquad \eta=\eta_b \tag{3.21}$$

Similarly ψ is obtained as

$$\psi = \frac{-\left(x_\eta x_{\eta\eta} + y_\eta y_{\eta\eta}\right)}{\left(x_\eta^2 + y_\eta^2\right)} \qquad \text{at} \qquad \xi = \xi_b \tag{3.22}$$

The values of ϕ, ψ in the interior are obtained by linearly interpolating ϕ in η direction and ψ in ξ direction. All the first and second derivatives in 3.21 and 3.22 are approximated using central difference.

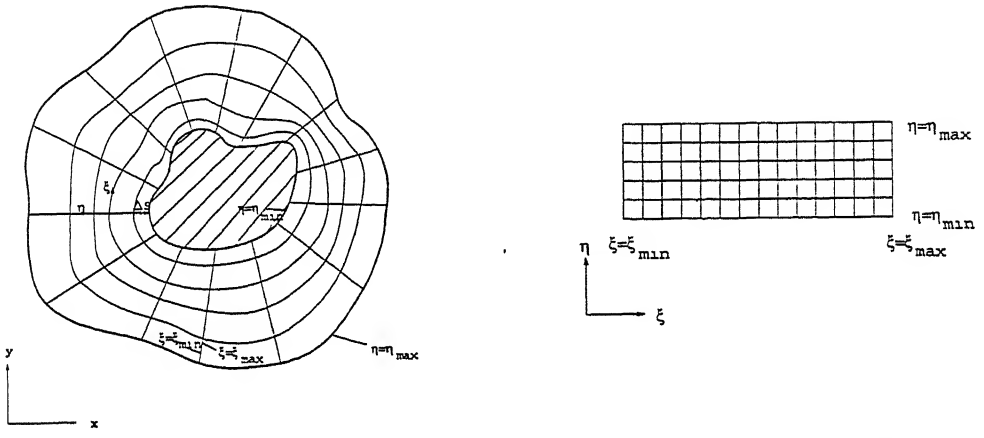
The evaluation of the functions ϕ, ψ is equivalent to constructing a local curve-fit to the arc length between the pre-assigned boundary points. The interpolation extends the range of curve-fit. Anderson and Steinbrenner [1] proved that the Thomas-Middlecoff formulation does not completely reflect the curvature effect of the boundary for interior points near boundaries. However, the Thomas-Middlecoff method gives satisfactory results for most of the problems. Also it converges quickly compared to Sorenson and Steger method which will be discussed in the next section. If the boundary points are placed at equal intervals, then the grid generated by Thomas-Middlecoff method and Laplace equations are same.

3.3.3 Sorenson Method

For some applications it is desirable to have grids that are orthogonal at the boundaries, for then the boundary conditions reduce to simple equations and certain terms may vanish. With nonorthogonal grids, the boundary conditions may transform into cumbersome equations in the computational domain, and may induce loss of numerical precision.

In 1979, Steger and Sorenson [8] developed a method to calculate the source terms automatically so that the intersection angle between the transverse coordinate lines with the boundaries, and the spacing between the boundaries and their adjacent coordinate, line can be controled effectively . This

essentially needs a judicious choice of the source functions P and Q in Equations 3.11 and 3.12. In this method, the spacing between the boundary and the adjacent grid line is specified. Also specified is the angle between the boundary line and its transverse lines. Using these constraints, the values of P and Q at the boundaries are determined by a prescribed procedure. Knowing these values at the boundaries, the P, Q values in the interior domain can be determined by interpolation. After that, the Equations 3.11 and 3.12 are used to generate the grid.



(a) physical domain

(b) computational domain

Figure 3.3: Sorenson Method

Suppose, the spacing and the intersection angle are to be controlled at $\eta - constant$ boundaries ($\eta = \eta_{min}$ and $\eta = \eta_{max}$). The P and Q values in the interior are chosen as

$$P(\xi, \eta) = P_{\eta_{min}}(\xi) \exp\{-a(\eta - \eta_{min})\} + P_{\eta_{max}}(\xi) \exp\{-b(\eta_{max} - \eta)\} \tag{3.23}$$

$$Q(\xi, \eta) = Q_{\eta_{min}}(\xi) \exp\{-c(\eta - \eta_{min})\} + Q_{\eta_{max}}(\xi) \exp\{-d(\eta_{max} - \eta)\} \tag{3.24}$$

where a, b, c, d are some positive constants, whose value has to be specified.

The first constraint imposed on the grid system is that the spacing along $\xi - constant$ lines between the boundary $\eta = \eta_b$ and its adjacent grid line is specified. Let this desired spacing be $\Delta s \mid_{\eta=\eta_b}$ then

$$\Delta s = \left[(\Delta x)^2 + (\Delta y)^2\right]^{\frac{1}{2}} \ at \ \eta = \eta_b$$

where Δx and Δy are the differences in x and y over the interval. When Δx and Δy approach zero, the differential relationship is

$$ds = \left[(dx)^2 + (dy)^2\right]^{\frac{1}{2}} \qquad \text{at} \qquad \eta = \eta_b$$

Using the transformation relations

$$ds = \left[(x_\xi d\xi + x_\eta d\eta)^2 + (y_\xi d\xi + y_\eta d\eta)^2\right]^{\frac{1}{2}} \qquad \text{at} \qquad \eta = \eta_b$$

Since ξ is constant along the interval under consideration, the above equation reduces to

$$ds = \left[\left(x_\eta^2 + y_\eta^2\right)^{\frac{1}{2}} d\eta\right] \qquad \text{at} \qquad \eta = \eta_b \, .$$

or equivalently

$$s_\eta = \left(x_\eta^2 + y_\eta^2\right)^{\frac{1}{2}} \qquad \text{at} \qquad \eta = \eta_b \, . \tag{3.25}$$

s_η is specified along the boundary $\eta = \eta_b$.

The second constraint imposed on the grid system is that the angle of intersection between the transverse grid lines and the boundary is specified. Let the angle be $\theta \mid_{\eta=\eta_b}$. Then from dot product definition

$$\nabla\xi \, . \, \nabla\eta = \mid \nabla\xi \mid \mid \nabla\eta \mid \cos\theta$$

Expanding this equation and transforming it to the computational domain we get

$$-x_\eta x_\xi - y_\eta y_\xi = \left(x_\eta^2 + y_\eta^2\right)^{\frac{1}{2}} \left(x_\xi^2 + y_\xi^2\right)^{\frac{1}{2}} \cos\theta \tag{3.26}$$

Similarly from the cross product definition

$$\nabla\xi \times \nabla\eta = |\nabla\xi||\eta| \sin\theta$$

or

$$-x_\eta y_\xi + y_\eta x_\xi = \left(x_\eta^2 + y_\eta^2\right)^{\frac{1}{2}} \left(x_\xi^2 + y_\xi^2\right)^{\frac{1}{2}} \sin\theta \tag{3.27}$$

Solving Equations 3.26 and 3.27 for x_η and y_η and using Equation 3.25 we get

$$x_\eta = s_\eta \frac{(-x_\xi \cos\theta - y_\xi \sin\theta)}{\left(x_\xi^2 + y_\xi^2\right)^{\frac{1}{2}}} \quad \text{at} \quad \eta = \eta_b\,. \tag{3.28}$$

$$y_\eta = s_\eta \frac{(-y_\xi \cos\theta + x_\xi \sin\theta)}{\left(x_\xi^2 + y_\xi^2\right)^{\frac{1}{2}}} \quad \text{at} \quad \eta = \eta_b\,. \tag{3.29}$$

Both θ and s_η is specified at $\eta = \eta_b$ and x_ξ, y_ξ can be calculated by differencing the x, y coordinates of the chosen boundary points. So x_η, y_η can be calculated at $\eta = \eta_b$ from the above two equations.

With similar procedure on $\xi = \xi_b$ boundaries, we can get x_ξ, y_ξ as

$$x_\xi = s_\xi \frac{(-x_\eta \cos\theta + y_\eta \sin\theta)}{\left(x_\eta^2 + y_\eta^2\right)^{\frac{1}{2}}} \quad \text{at} \quad \xi = \xi_b\,. \tag{3.30}$$

$$y_\xi = s_\xi \frac{(-y_\eta \cos\theta - x_\eta \sin\theta)}{\left(x_\eta^2 + y_\eta^2\right)^{\frac{1}{2}}} \quad \text{at} \quad \xi = \xi_b\,. \tag{3.31}$$

where $s_\xi = \left(x_\xi^2 + y_\xi^2\right)^{\frac{1}{2}}$ is the spacing between $\xi = \xi_b$ boundary and its adjacent grid line which needs to be specified.

The four positive constants a, b, c, d in the Equation 3.23 and 3.24 determine the extend by which the control functions at the boundaries affect the interior. These constants are chosen such that the effect of control functions of the one boundary is vanishingly small at the opposite boundary. so only one term will remain in the right hand side of the Equations 3.23 and 3.24 along

$\eta = \eta_b$ boundaries. Now substituting into the Equations 3.11 and 3.12 we get

$$P(\xi, \eta) = \frac{y_\eta R_1 - x_\eta R_2}{|J|} \quad \text{at} \quad \eta = \eta_b \tag{3.32}$$

$$Q(\xi, \eta) = \frac{-y_\xi R_1 + x_\xi R_2}{|J|} \quad \text{at} \quad \eta = \eta_b \tag{3.33}$$

where

$$R_1 = \frac{-(\alpha x_{\xi\xi} - 2\beta x_{\xi\eta} + \gamma x_{\eta\eta})}{|J|^2} \quad \text{at} \quad \eta = \eta_b \tag{3.34}$$

$$R_2 = \frac{-(\alpha y_{\xi\xi} - 2\beta y_{\xi\eta} + \gamma y_{\eta\eta})}{|J|^2} \quad \text{at} \quad \eta = \eta_b \tag{3.35}$$

We will obtain the same Equations 3.33 - 3.35 at $\xi = \xi_b$ if the grid lines were to be controlled at $\xi = \xi_b$.

For boundaries $\eta = \eta_b$, The $\xi-$ derivatives x_ξ, y_ξ, $x_{\xi\xi}$, $y_{\xi\xi}$ are simply calculated from central difference approximation of the chosen boundary points and the first derivatives x_η and y_η are calculated using the Equations 3.28 and 3.29. The cross derivatives $x_{\xi\eta}$, $y_{\xi\eta}$ can be computed by differencing x_η and y_η. The double derivatives $x_{\eta\eta}$ and $y_{\eta\eta}$ are computed by differencing the current approximation of the x, y field using

$$x_{\eta\eta}|_{j=1} = \frac{-7x|_{j=1} + 8x|_{j=2} - x|_{j=3}}{2(\Delta\eta)^2} - \frac{3x_\eta|_{j=1}}{\Delta\eta}$$

$$y_{\eta\eta}|_{j=1} = \frac{-7y|_{j=1} + 8y|_{j=2} - y|_{j=3}}{2(\Delta\eta)^2} - \frac{3y_\eta|_{j=1}}{\Delta\eta}$$

$$x_{\eta\eta}|_{j=jmax} = \frac{-7x|_{j=jmax} + 8x|_{j=jmax-1} - x|_{j=jmax-2}}{2(\Delta\eta)^2} + \frac{3x_\eta|_{j=jmax}}{\Delta\eta}$$

$$y_{\eta\eta}|_{j=jmax} = \frac{-7y|_{j=jmax} + 8y|_{j=jmax-1} - y-|_{j=jmax-2}}{2(\Delta\eta)^2} + \frac{3y_\eta|_{j=jmax}}{\Delta\eta}$$

where the grid node i, j corresponds to

$$\xi(i, j) = (i-1)\ \Delta\xi$$

$$\eta(i, j) = (j-1)\ \Delta\eta$$

Because $x_{\eta\eta}$ and $y_{\eta\eta}$ need t he knowledge of the x, y field, this method needs an iterative solution procedure. The double derivatives $x_{\eta\eta}$ and $y_{\eta\eta}$ keep changing with iterations of the the solution until convergence is achieved.

The source terms can be quite large and if in every iteration, values calculated from Equations 3.33 and 3.33 are used exactly, instabilities can result. So under relaxation becomes essential in the iterative procedure. The source terms at t he boundary are updated by using the following expressions.

$$P^{(n+1)} = P^{(n)} + w_P \left[J^{-1} \left(y_\eta R_1 - x_\eta R_2\right)^{(n+1)} - P^{(n)}\right]$$
$$Q^{(n+1)} = Q^{(n)} + w_Q \left[J^{-1} \left(-y_\xi R_1 + x_\xi R_2\right)^{(n+1)} - Q^{(n)}\right]$$

where n is the iteration level and the weight functions w_P, w_Q are quite small (0.02 to 0.06).

For sharp corners, occuring along the boundaries, the computed source terms have to be replaced by an average of the source terms on either side of the corner.

The above method, as discussed by Steger and Sorenson [8], can control the grid lines over only two boundaries. To control the grid lines over all the boundaries. instead of control functions of the form Equation 3.23 and 3.24, we used

$$\begin{aligned} P(\xi, \eta) &= P_{\xi_{min}}(\eta)\ \eta(1-\xi)(1-\eta)\ \exp\{-a(\xi-\xi_{min})\} \\ &+ P_{\xi_{max}}(\eta)\ \xi\eta(1-\eta)\ \exp\{-b(\xi_{max}-\xi)\} \\ &+ P_{\eta_{min}}(\xi)\ \xi(1-\xi)(1-\eta)\ \exp\{-c(\eta-\eta_{min})\} \\ &+ P_{\eta_{max}}(\xi)\ \xi\eta(1-\xi)\ \exp\{-d(\eta_{max}-\eta)\} \end{aligned} \tag{3.36}$$

$$\begin{aligned} Q(\xi, \eta) &= Q_{\xi_{min}}(\eta)\ \eta(1-\xi)(1-\eta)\ \exp\{-a(\xi-\xi_{min})\} \\ &+ Q_{\xi_{max}}(\eta)\ \xi\eta(1-\eta)\ \exp\{-b(\xi_{max}-\xi)\} \\ &+ Q_{\eta_{min}}(\xi)\ \xi(1-\xi)(1-\eta)\ \exp\{-c(\eta-\eta_{min})\} \\ &+ Q_{\eta_{max}}(\xi)\ \xi\eta(1-\xi)\ \exp\{-d(\eta_{max}-\eta)\} \end{aligned} \tag{3.37}$$

In many geometries, it is impossible to maintain both orthogonality and small spacing between grid lines in all the boundaries. If we use the Sorenson method for these geometries, the solution will not converge or a bad grid will result even after a thousands of iterations.

In our experience, we found that instead of calculating both source terms P and Q in all the boundaries, calculating P in $\xi - constant$ boundaries and Q in $\eta - constant$ boundaries yields good results. Let us call this method as Modified Sorenson method. The evaluation of source terms at the boundaries is the same as that of the Sorenson method, except that only one source term is determined from each boundary. The control functions in the interior are calculated from

$$P(\xi, \eta) = P_{\xi_{min}}(\eta) \exp\{-a(\xi - \xi_{min})\} + P_{\xi_{max}}(\eta) \exp\{-b(\xi_{max} - \xi)\} \tag{3.38}$$

$$Q(\xi, \eta) = Q_{\eta_{min}}(\xi) \exp\{-c(\eta - \eta_{min})\} + Q_{\eta_{max}}(\xi) \exp\{-d(\eta_{max} - \eta)\} \tag{3.39}$$

As we know that the source term P controls the spacing in the ξ direction and intersection angles at the $\eta - constant$ boundaries and Q controls the spacing in the η direction and intersection angles at the $\xi - constant$ boundaries. the Modified Sorenson method stresses more on spacing control than control of intersection angles. Even so the Modified Sorenson method generally delivers a near-orthogonal grid in the boundaries.

In the Modified Sorenson method, to save computations, the control functions are calculated only once instead of recomputing $x_{\eta\eta}$ in $\eta - constant$ boundaries and $x_{\xi\xi}$ in $\xi - constant$ boundaries are updated after each Gauss seidel iteration sweep

3.4 Orthogonal Grids - Moving Boundary Points Method

Without using any control functions, orthogonality can be achieved at the boundaries by suitably choosing the boundary points. But there is no direct way of choosing the correct grid points without knowing the interior grid points which themselves depend upon the boundary grid point distribution. So the boundary points are chosen, and then corrected iteratively to obtain a locally orthogonal grid at the boundaries. This method, given below, is called the Moving Boundary Points Method.

To be orthogonal, the grid should satisfy

$$x_\xi x_\eta + y_\xi y_\eta = 0 \tag{3.40}$$

at that point. First with some assumed boundary point distribution, interior grid points are calculated by solving the Laplace Equations 3.3 and 3.4. To find out new boundary points on $\xi - constant$ boundaries, if $\mid x_\eta \mid < \mid y_\eta \mid$ then y_ξ is calculated from the orthogonality condition 3.40 i.e.

$$y_\xi = -\frac{x_\xi x_\eta}{y_\eta}$$

where the quantities on the right are estimated from the previous assumed x, y field. If $\mid y_\eta \mid < \mid x_\eta \mid$, then x_ξ is similarly computed

$$x_\xi = -\frac{y_\xi y_\eta}{x_\eta} .$$

This procedure avoids the difficulty caused if either x_η or y_η is zero. Suppose, if x_ξ is calculated, then the x coordinate of the new boundary point is recomputed from the finite difference form of x_ξ keeping the interior grid nodes at previous values. Then y is computed from the equations of the boundary. Suppose if y_ξ is calculated first, then y is calculated from the difference from of y_ξ and x is calculated from the equations of the boundaries.

It is necessary to slow down the corrections in the boundary points near the corners. Otherwise some of the points in one boundary can move to the other boundary and the mapping will not be one to one anymore. So an under relaxation factor which is a function of the distance of the point from its nearest corner is used. Thus the new coordinate is calculated using

$$x_{new} = x_{pre} + \left(1 - \exp\left\{-k\left(\frac{d}{d_n}\right)\right\}\right)[x_{cal} - x_{pre}]$$

where

- $_{pre}$ - represents previous value.
- $_{cal}$ - represents the value calculated from the finite difference form.
- d - is the distance between the grid point and its nearest corner.
- d_n - can be chosen as one third of the distance between two nearest cor
- k - is some constant in the range 1–4.

The y coordinate corresponding to the calculated x is calculated from the equations of the boundary.

For η – *constant* boundaries, a similar procedure is used. This whole procedure is repeated until the desired grid is obtained. This method is computationally very expensive compared to other methods. We have implemented this method only for a particular section.

Chapter 4

RESULTS AND DISCUSSION

While writing codes for grid generation, we are most concerned about boundary orthogonal grids which will make implementation of boundary conditions simple and in the clustering of grid points near the boundaries to capture the boundary layer. However there are many other considerations which need to be included to certify a grid as 'good'. It is difficult to identify a single method which is the best of all because the requirements of the grids vary from problem to problem.

Grids generated by various methods for the same domain can be compared using the parameters such as skewness, aspect ratio, adjacent cell ratio, nearness parameter, etc. Even then if the parameters are within certain limits, all the grids are acceptable: at the same time there is no unique general criteria with which to certify a particular grid the best.

A method which gives a better value for a particular property may not give a better result for a second property . In such cases, a greater relative importance can be given to certain parameters depending upon the problem in which the grid is to be employed. For examples, for viscous flow problems, the nearness parameter can be very important since it ensures relatively more

number of grid points near the boundaries.

In all the methods, the interior grid point distribution depends upon boundary grid point distribution which is entirely specified by the user. So the intitutive skill in the choice of boundary points also comes in to play.

The quality of the grid generated by the Sorenson and Modified Sorenson method depends upon the exponential decay factor, the spacing between the boundaries and their adjacent grid lines, and the intersection angle between the grid lines at the boundaries, which are also specified by the user.

To study the effectiveness of various methods, grids were developed for the following sections.

1. Circular section
2. Converging section
3. L - section
4. Quarter portion of a super ellipse
5. Channel section with a downward slope at the middle

Grids were developed for both uniform as well as nonuniform boundary point distribution. Except for the circular section, exponential functions were chosen to generate nonuniform boundary point distribution. The various parameters viz skewness, aspect ratio, adjacent cell ratio, nearness parameter are computed for all the grids and were used for comparison. These parameters are defined in section 2.6 of this thesis.

4.1 Case 1: Circular Section

The figures 4.1(a)-(e) show grids for a circular section with equally spaced boundary points. The figures show the grids obtained by (a) Transfinite interpolation (b) Laplace method (c) Thomas-Middlecoff method (d) Sorenson and (e) Modified Sorenson method, respectively. The salient features of the various methods emerge from a comparison of these figures. Transfinite interpolation gives highly skewed cells at the corners, The Laplace grid (Fig 4.1(b)) has equally spaced and orthogonal grid lines except near the boundaries from where the grid lines were repelled due to concave curvature of the boundary. With equally spaced boundary points, Thomas-Middlecoff method 4.1(c) gives the same results as that of the Laplace system. The Sorenson method 4.1(d) gives a near-orthogonal grids in the boundaries. Modified Sorenson method tries to get the specified spacing at the boundaries. For both Sorenson and Modified Sorenson method, half of the spacing between the boundaries and their adjacent grid lines of the Transfinite grid is specified as the spacing. Also the intersection angle at the boundary is chosen as 90°. The exponential decay factor is taken as $0.5/\Delta\eta$.

Figures 4.2 (a)-(e) shows the plot of cumulative percentage of the grid cells Vs various parameters. All the parameters - skewness, adjacent cell ratio, aspect ratio, intersection angles between grid lines, nearness parameter - were taken at the abscissa. The plots show what percentage of the grid cells are above or below that value on the horizontal axis. Adjacent cell ratio and aspect ratio are nearly same for all methods. The Laplace method and Thomas-Middlecoff give less skewed grids while Sorenson gives the most skewed grid. This is due to the cells near the corners. The Modified Sorenson performance regarding skewness is somewhat better than that of the Sorenson method. The nearness parameter is nearly same for all methods.

Figure 4.3(b) shows the grids generated by iteratively moving the boundary points to get boundary orthogonal grids, as discussed in section 3.4. The boundary points were moved towards the corners. The figures 4.3(a) and 4.3(c) show the grids generated by the Transfinite interpolation and the Thomas-Middlecoff method for the same boundary points as in figure 4.3(b). Both Sorenson and Modified Sorenson methods give bad grids for this boundary point distribution. For these results, the parameters were plotted in the figures 4.4(a)-(e).

From these figures, it can be seen that Transfinite interpolation gives highly skewed grids at the corners. Adjacent cell ratio is minimum for the Laplace method which is due to its strong smoothing property. However, although the Laplace method gives less skewed, smooth grids, its grid lines are far away from the boundary. Hence Laplace method is not suitable for viscous flow calculations. The Thomas-Middlecoff gives a fine grid at the boundaries, suitable for resolving the boundary layer. This is undoubtably the 'best' grid.

4.2 Case 2: Converging Section

The figures 4.5(a) and 4.7(e) show grids generated for a converging section with two types of boundary point distribution. The figures 4.6 and 4.8 show the properties plots for this grids. With equally spaced boundary points, all methods are doing well. For both Sorenson and Modified Sorenson method, the exponential decay factor is chosen as $1/\Delta\eta$. and the spacing is specified as 0.75 times the spacing in the Transfinite grid. It can be easily seen from the figures that due to higher value of exponential decay factor only few grid lines near the boundary were affected. Nearness parameter is equal to 1 for the grid generated by Transfinite interpolation. For other methods, it varies from 0.5 to 1.5.

Thomas-Middlecoff is doing well with exponentially spaced boundary Thomas-Middlecoff gives the results with closely spaced near-orthogonal grid lines at the boundaries. Transfinite interpolation itself gives a better grid than the Laplace, Sorenson and Modified Sorenson, which all tend to have large spacing near the boundaries.

4.3 Case 3: L - Section

The figures 4.9(a)-(e) and 4.11(a)-(e) show the grids generated for a right angled bend. The boundary discontinuity was propagated into the grids generated by Transfinite interpolation. For the equally spaced boundary points case, the Transfinite interpolation seems to be best for the others gives rather large grid spacing near the corners. But on the whole, the grids are compared in terms of skewness, aspect ratio etc.. In the exponentially spaced boundary points case, Thomas-Middlecoff does best in terms of the grid performance near the boundary with Transfinite interpolation coming second, the others showing unacceptable performance at the corners.

The figures 4.10 and 4.12 show the property plot for these grids. Transfinite interpolation gives grids with lower skewness. Sorenson method gives higher number of grid cells with slightly large skewness. Adjacent cell ratio is slightly better for Transfinite grids. With equally spaced boundary points, Transfinite and Laplace method gives smaller aspect ratio grids than Sorenson or Modified Sorenson method. But with exponential boundary point distribution, Thomas-Middlecoff and Transfinite interpolation give large number of grid cells with higher aspect ratio, but from figures 4.10(a)-(e) they give the best grids!. All this goes to show that it is difficult to judge the performance of the methods by quantitative parameters - the best judge of god grid may still be qualitative 'eye ball' estimate.

4.4 Case 4: Rectangular Channel Section with a Slope

The figures 4.13(a)-(e) and 4.15(a)-(e) show the grid generated for a channel with a downward slope at the middle. As before, we can see the discontinuity in the interior in the grid generated by Transfinite interpolation. Also the grid lines are away from the boundary near the concave portion of the boundary in the grid generated by Laplace method. Sorenson method tries to get orthogonal grid lines and hence make the grid lines curved which are vertical in other methods.

Figures 4.15(a)-(e) and 4.16(a)-(e) show the case of exponentially spaced boundary points. As usual, with exponentially spaced boundary points, Thomas-Middlecoff method does well. Transfinite interpolation also yields an acceptable grid except in terms of smoothness. The other methods fail in maintaining nearness at the boundary. However both Thomas-Middlecoff and Transfinite interpolation give rather large aspect ratios (figure 4.16(c)).

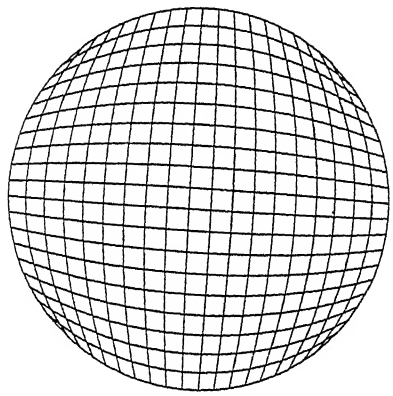
4.5 Case 5: Quarter Portion of a Super Ellipse

The figures 4.17(a)-(e) and 4.19(a)-(e) show the grids generated in a quarter portion of the superellipse $\left(\frac{x}{a}\right)^n + \left(\frac{y}{b}\right)^n = 1$ with $n = 3, \frac{b}{a} = 1.4$. The same illustration was used by Thomas and Middlecoff [11] to test their method.

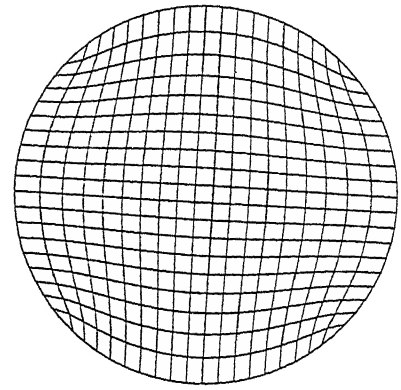
With equally spaced grid points in the boundaries, Transfinite, Laplace, Thomas-Middlecoff and Sorenson give similar results. While Modified Sorenson fails to give a good grid. With exponentially spaced boundary points Thomas-Middlecoff gives the best grid with evenly spaced cells close to the boundary. Transfinite interpolation also gives an acceptable grid while the Laplace tends to pull the grid lines away from the boundary.

4.6 Conclusions

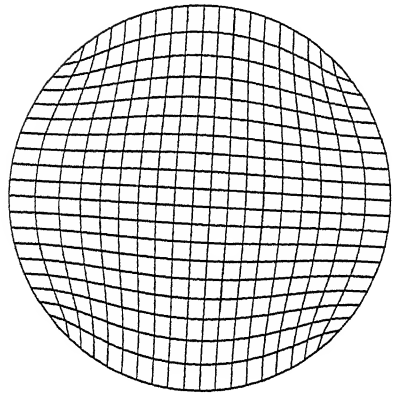
A number of methods of grid generation, Transfinite interpolation, Laplace, Thomas-Middlecoff, Sorenson, Modified Sorenson method, were used to generate H-type grid for a variety of configurations and their performance was evaluated. Special attention was paid to the ability of the method to deliver grid lines that followed the boundaries closely. This property has great relevance for grid generated for viscous computations in internal geometries. The performance of Thomas-Middlecoff was clearly the best with Transfinite interpolation coming second, although the grids of the later were not always smooth. The other methods were quite erratic, often failing badly.



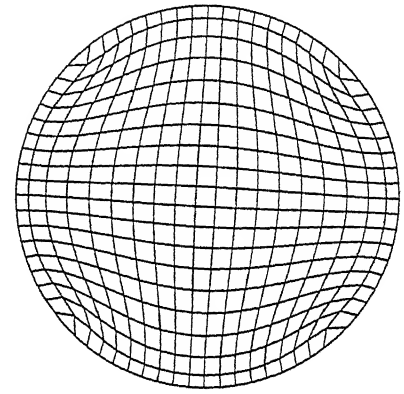
(a) Transfinite interpolation



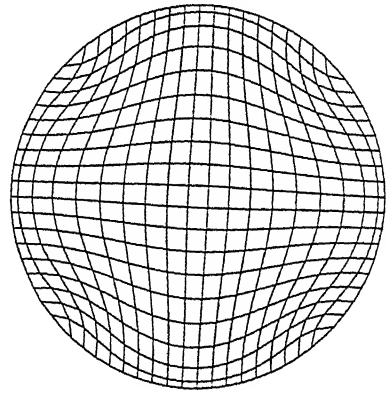
(b) Laplace method



(c) Thomas-Middlecoff Method



(d) Sorenson Method



(e) Modified Sorenson Method

Figure 4.1: Circular Section with Uniform Boundary Point Distribution

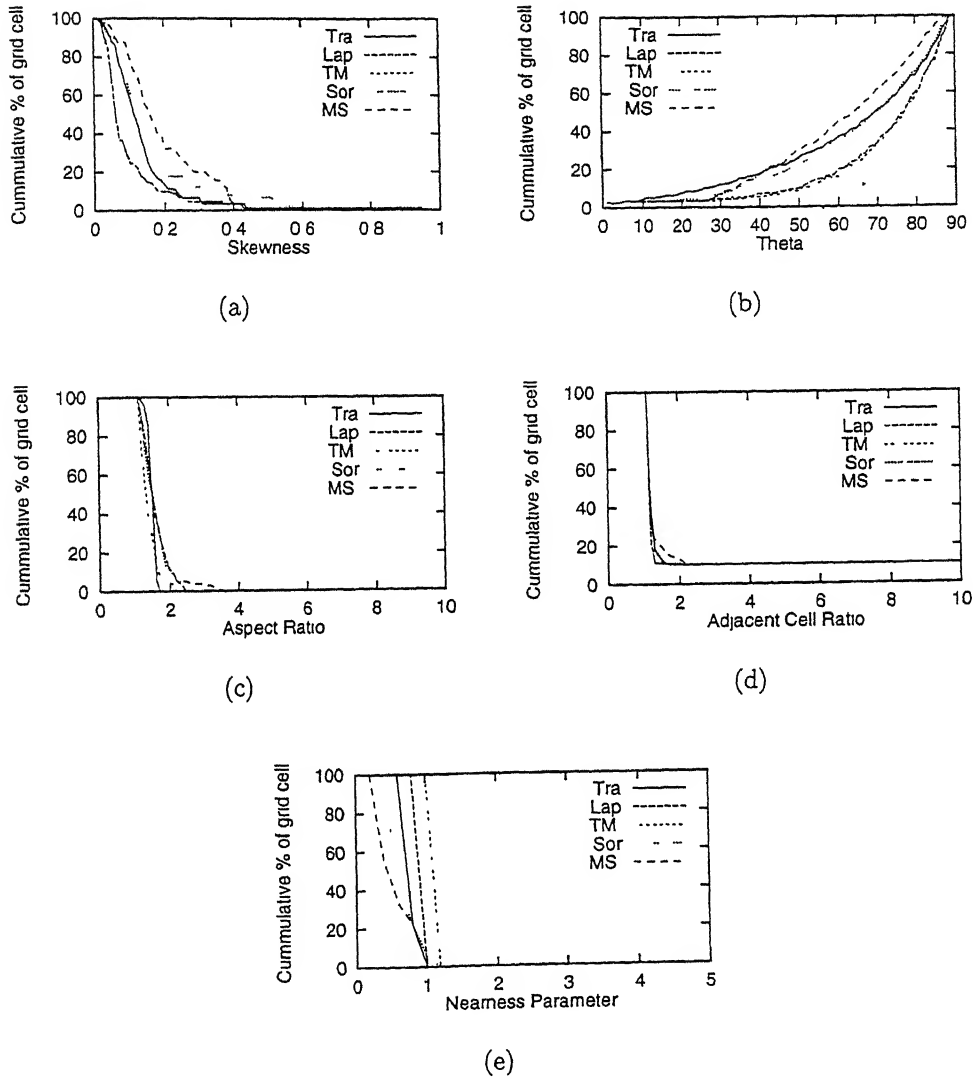
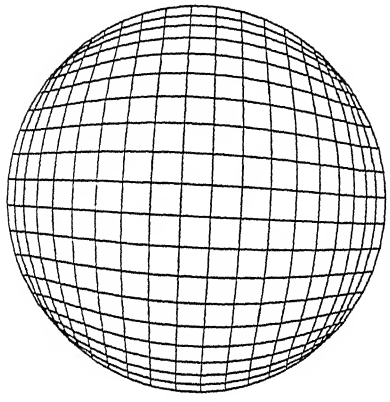
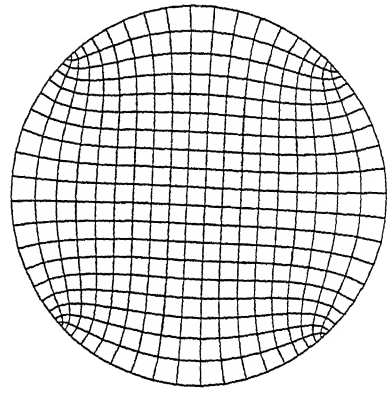


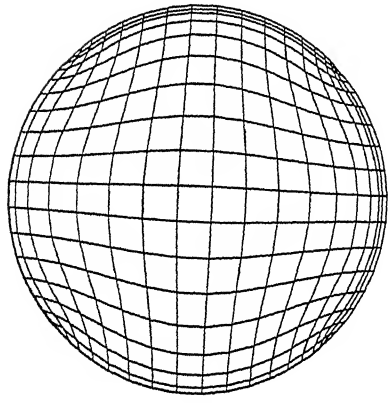
Figure 4.2: Parameters Plots for circular Section with Uniform Boundary Point Distribution



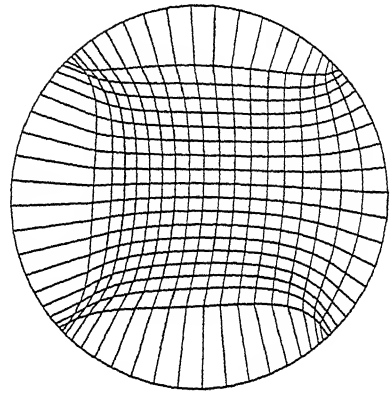
(a) Transfinite interpolation



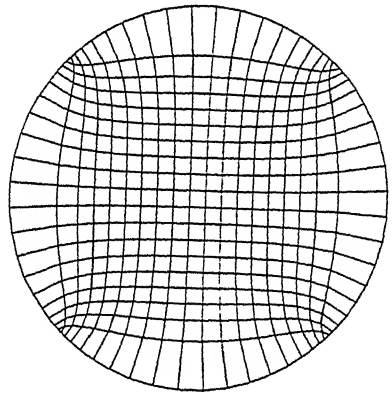
(b) Laplace method



(c) Thomas-Middlecoff Method



(d) Sorenson Method



(e) Modified Sorenson Method

Figure 4.3: Circular Section with Nonuniform Boundary Point Distribution

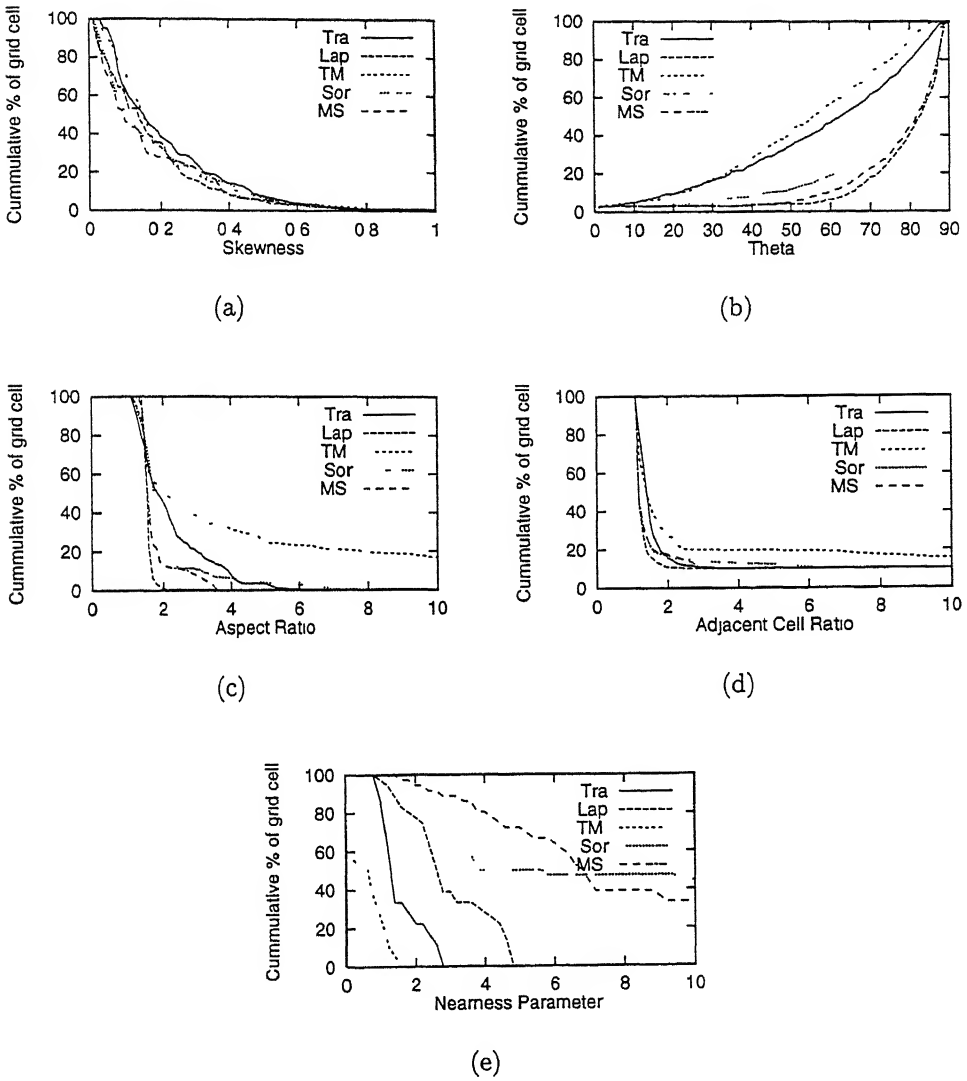
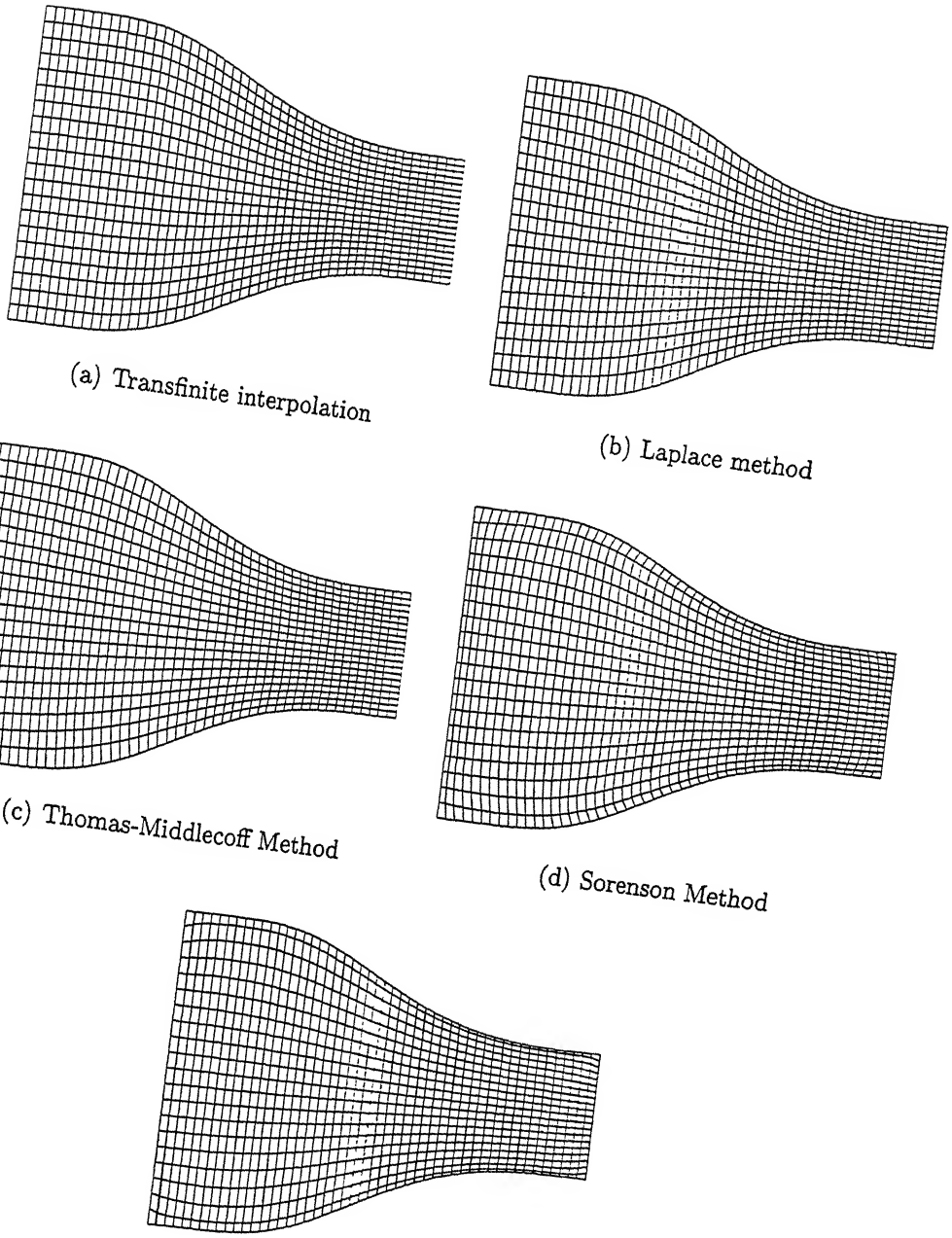


Figure 4.4: Parameters Plot for Circular Section with Nonuniform Boundary Point Distribution



(a) Transfinite interpolation

(b) Laplace method

(c) Thomas-Middlecoff Method

(d) Sorenson Method

(e) Modified Sorenson Method

Figure 4.5: Converging Section with Uniform Boundary Point Distribution

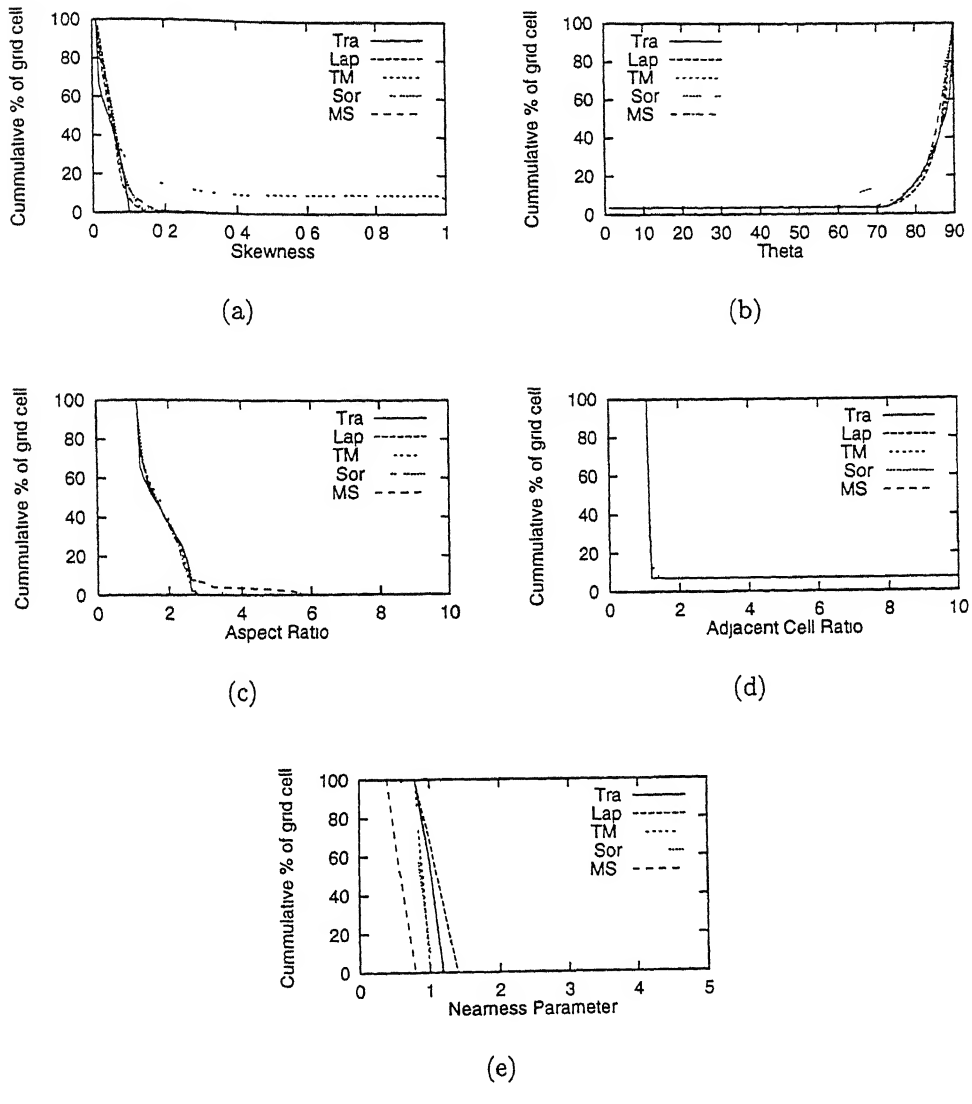
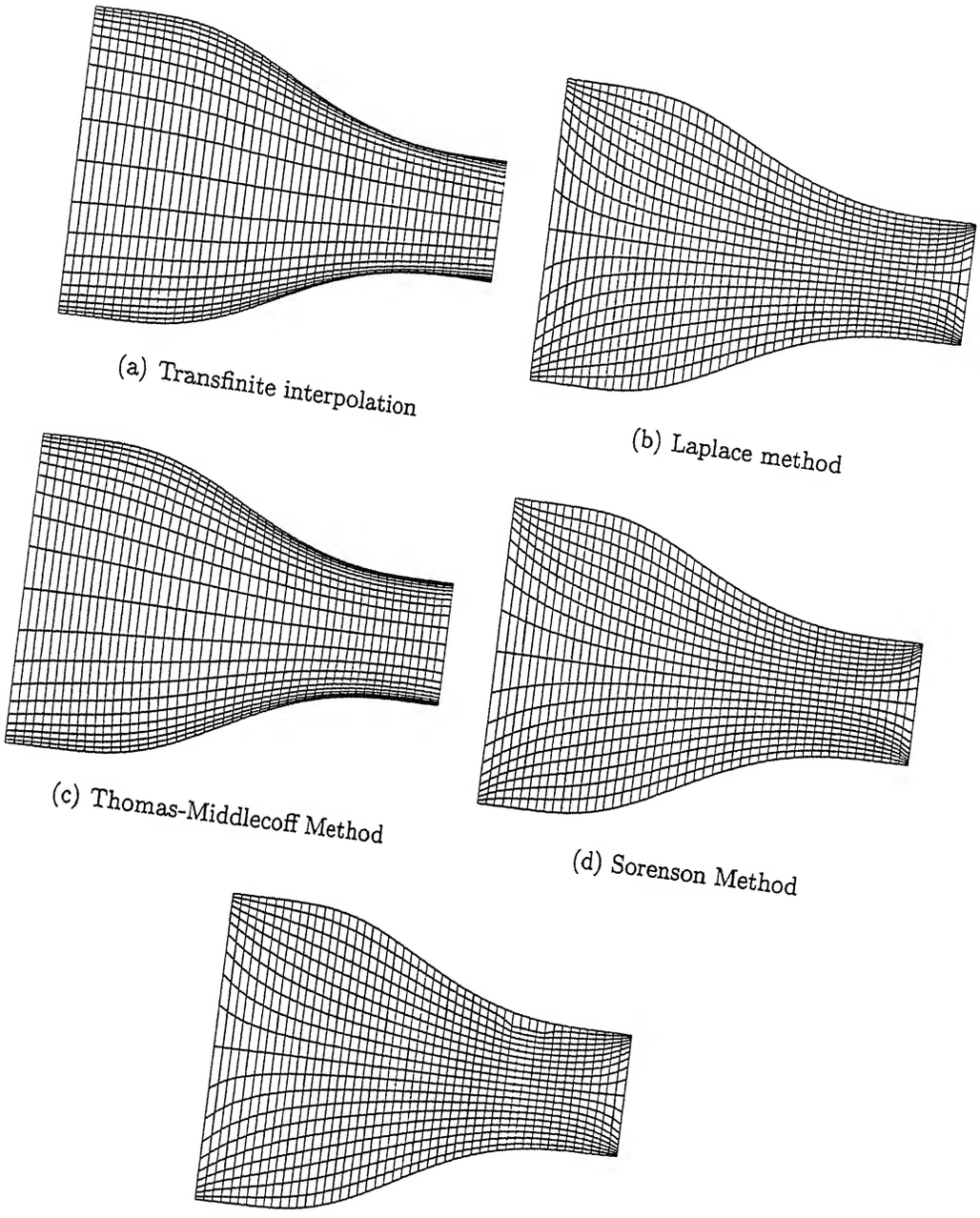


Figure 4.6: Parameters Plot for Converging Section with Uniform Boundary Point Distribution



(a) Transfinite interpolation

(b) Laplace method

(c) Thomas-Middlecoff Method

(d) Sorenson Method

(e) Modified Sorenson Method

Figure 4.7: Converging Section with Nonuniform Boundary Point Distribution

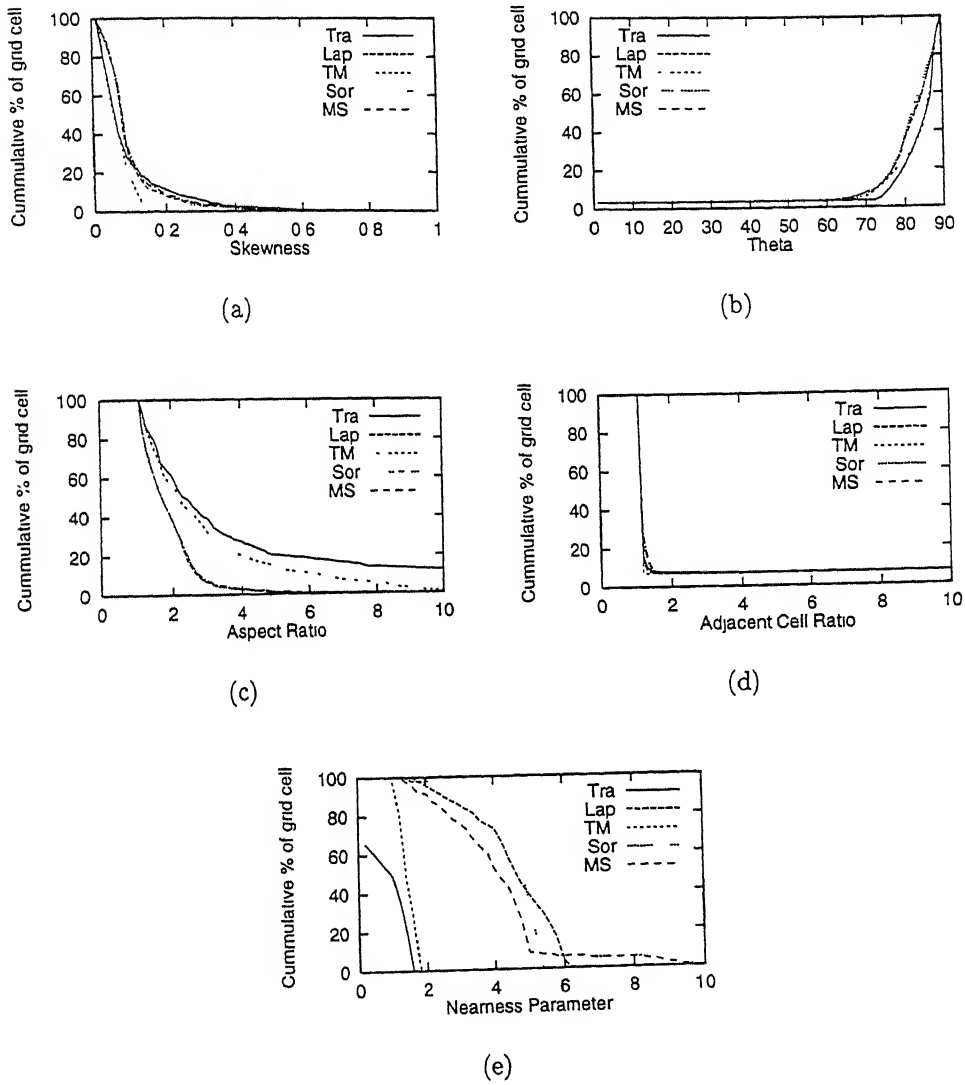
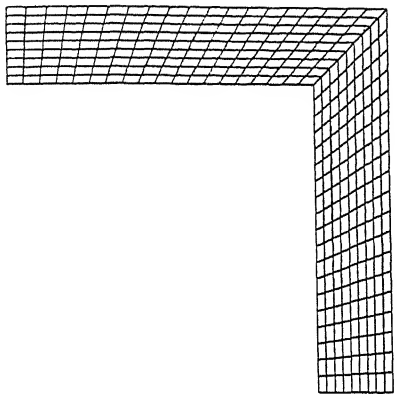
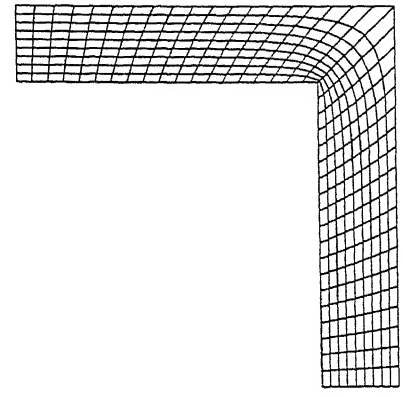


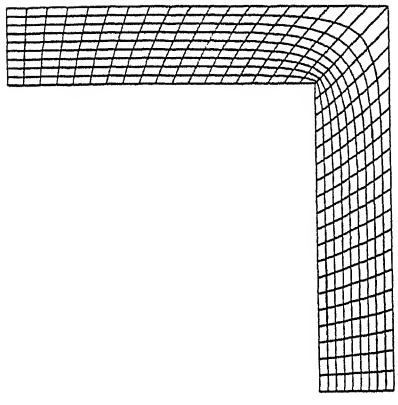
Figure 4.8: Parameters Plot for Converging Section with Nonuniform Boundary Point Distribution



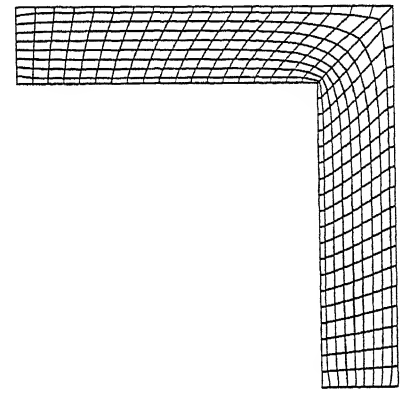
(a) Transfinite interpolation



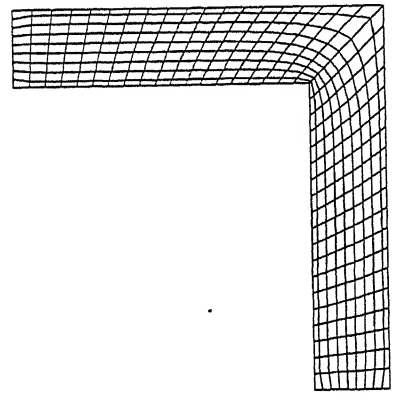
(b) Laplace method



(c) Thomas-Middlecoff Method



(d) Sorenson Method



(e) Modified Sorenson Method

Figure 4.9: L - Section with Uniform Boundary Point Distribution

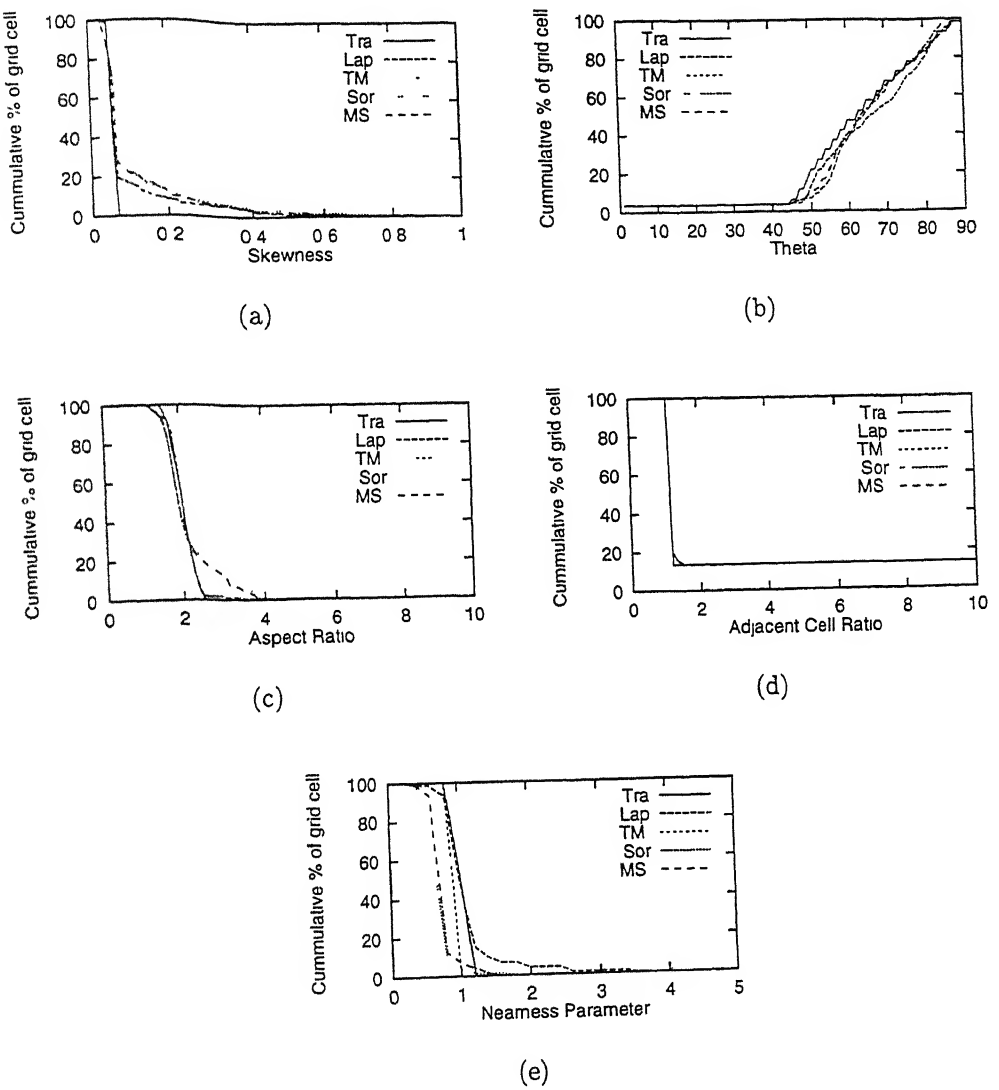
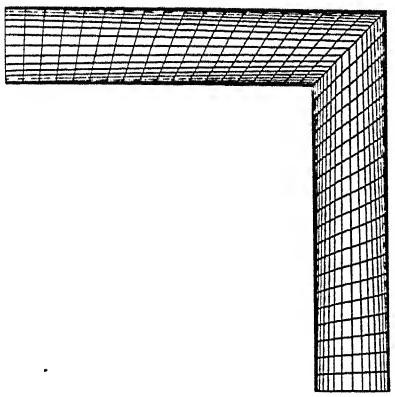
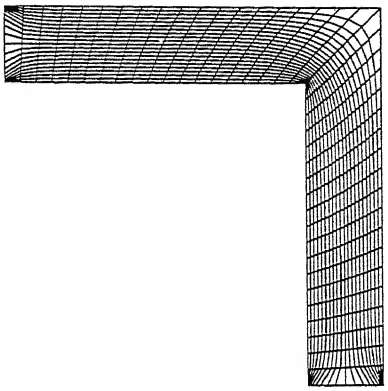


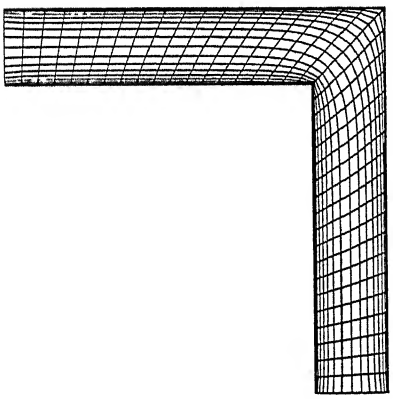
Figure 4.10: Parameters Plot for L - Section with Uniform Boundary Point Distribution



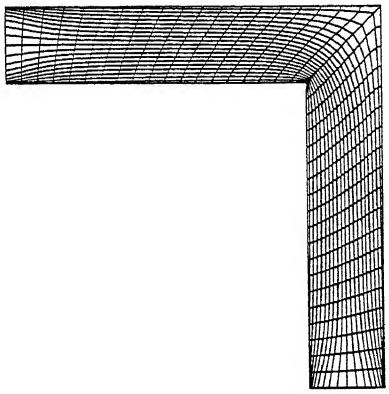
(a) Transfinite interpolation



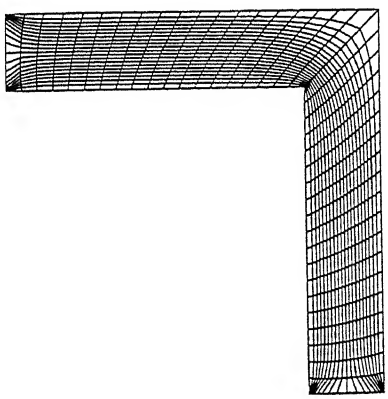
(b) Laplace method



(c) Thomas-Middlecoff Method



(d) Sorenson Method



(e) Modified Sorenson Method

Figure 4.11: L - Section with Nonuniform Boundary Point Distribution

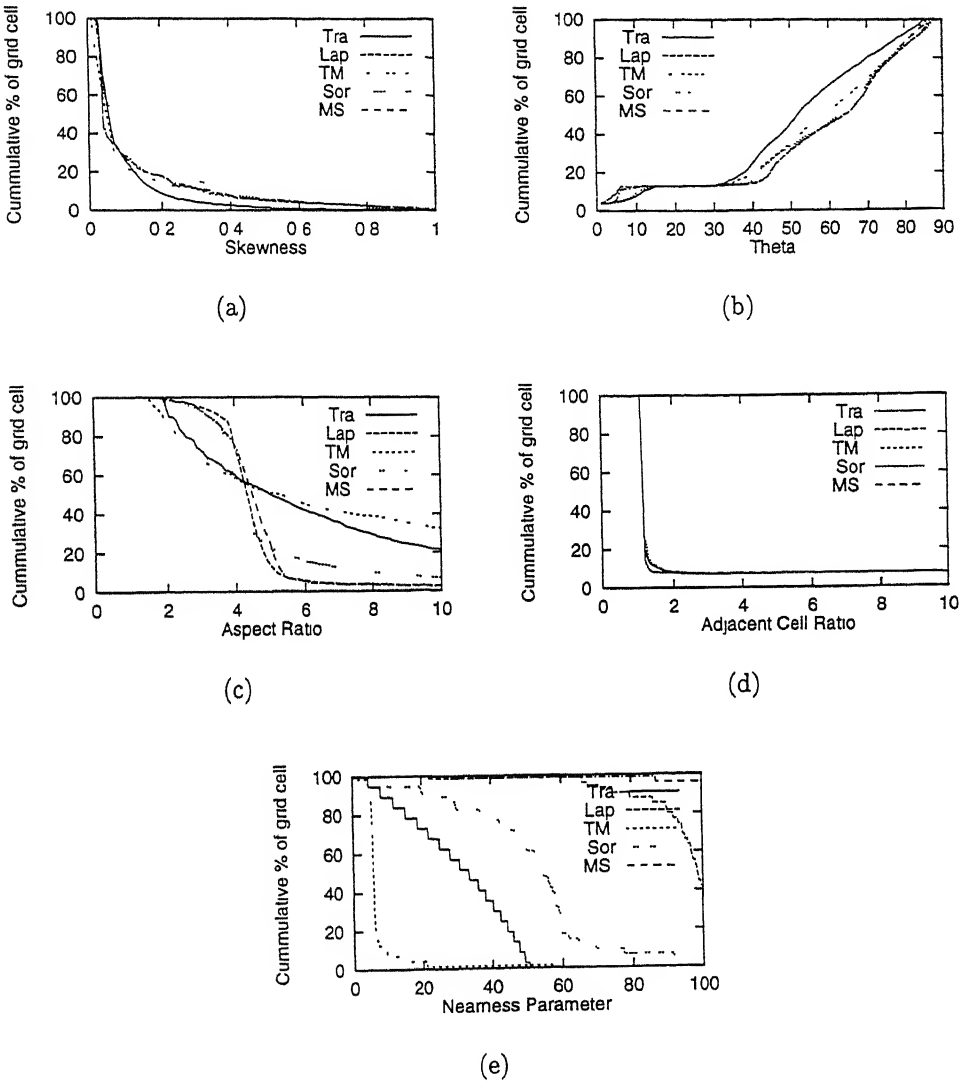
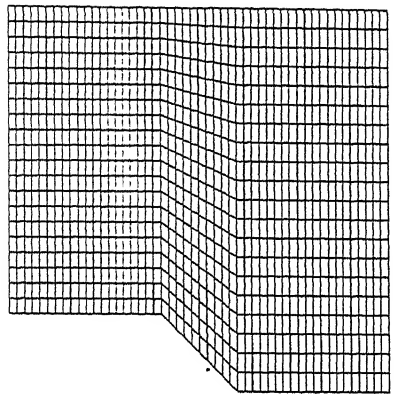
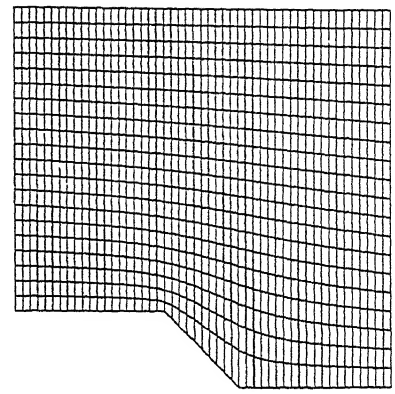


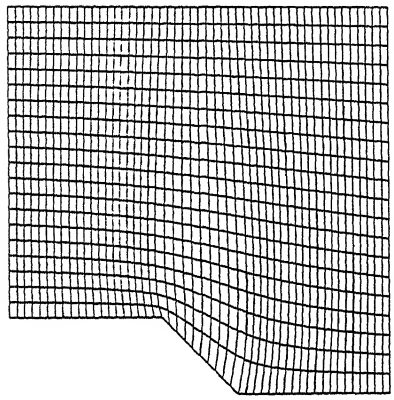
Figure 4.12: Parameters Plot for L - Section with Nonuniform Boundary Point Distribution



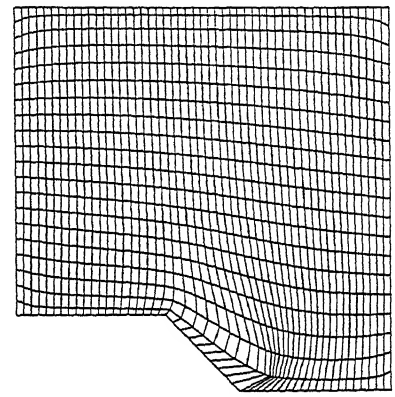
(a) Transfinite interpolation



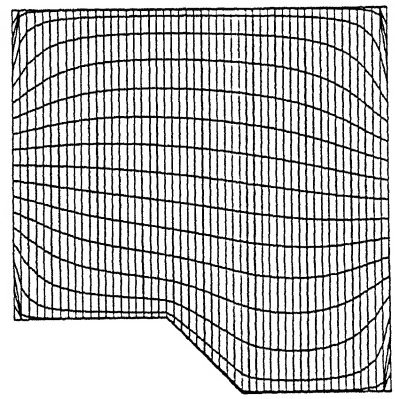
(b) Laplace method



(c) Thomas-Middlecoff Method



(d) Sorenson Method



(e) Modified Sorenson Method

Figure 4.13: Rectangular Channel Section with a Slope with Uniform Boundary Point Distribution

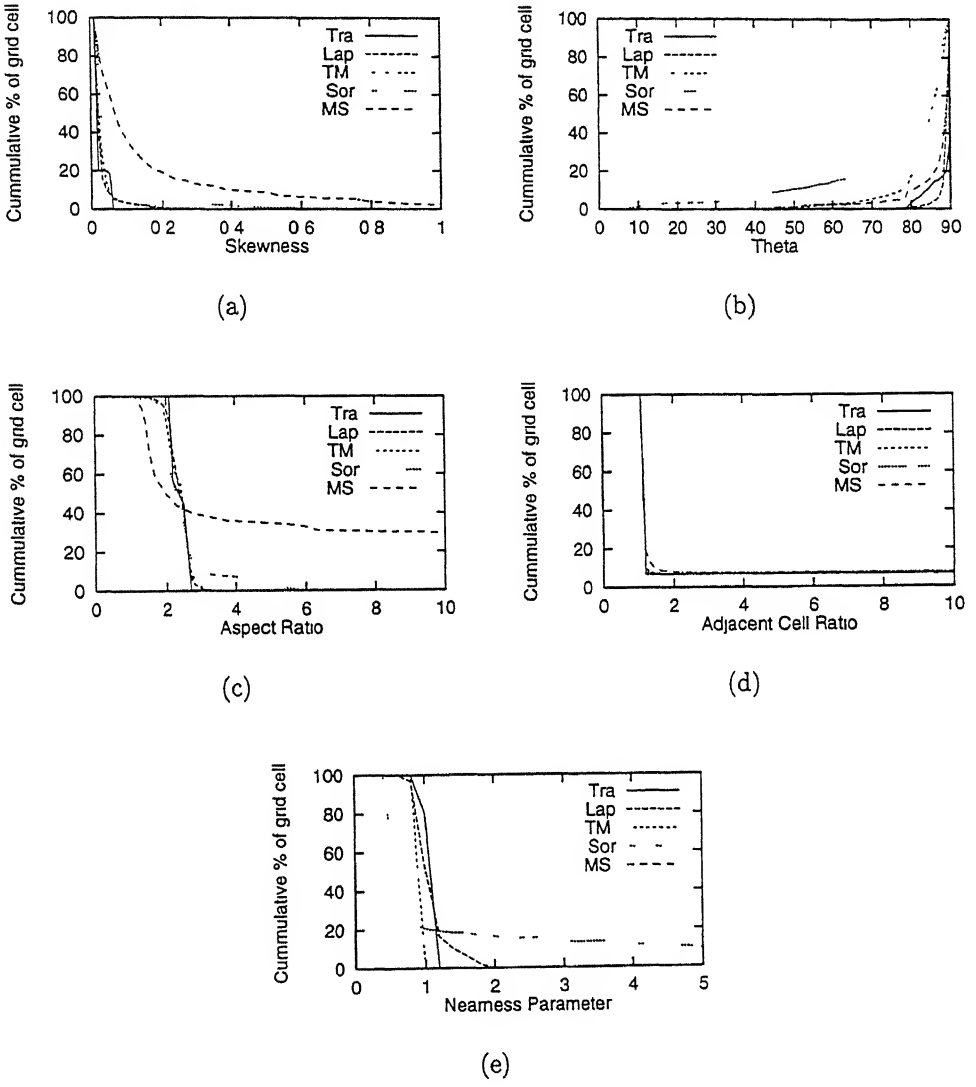
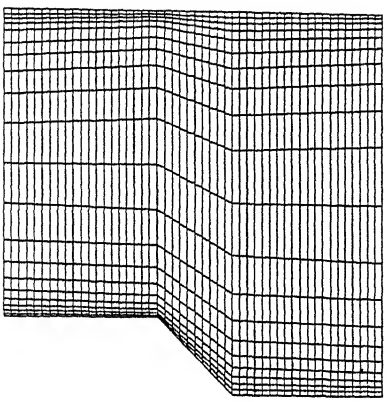
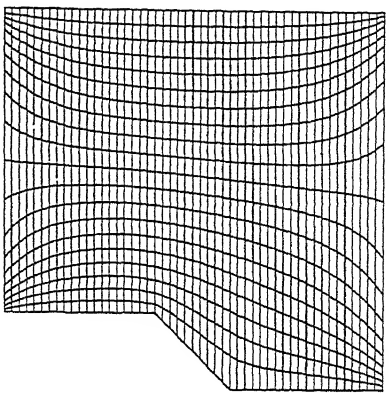


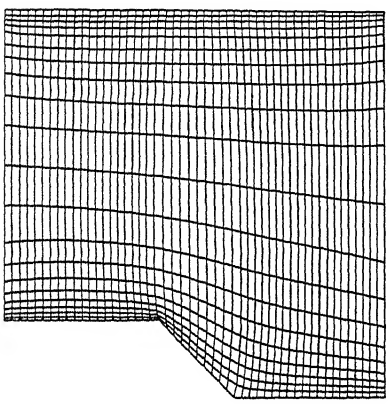
Figure 4.14: Parameters Plot for Rectangular Channel with a Slope with Uniform Boundary Point Distribution



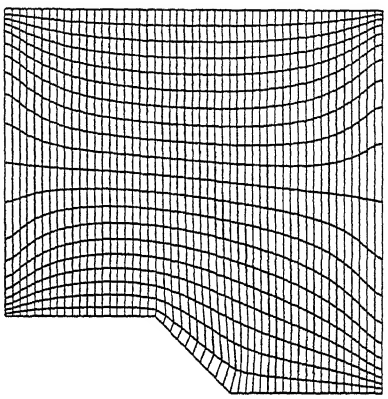
(a) Transfinite interpolation



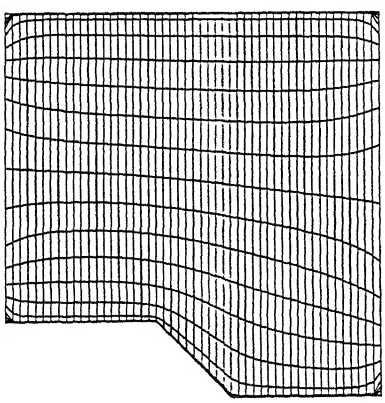
(b) Laplace method



(c) Thomas-Middlecoff Method



(d) Sorenson Method



(e) Modified Sorenson Method

Figure 4.15: Rectangular Channel with a Slope with Nonuniform Boundary Point Distribution

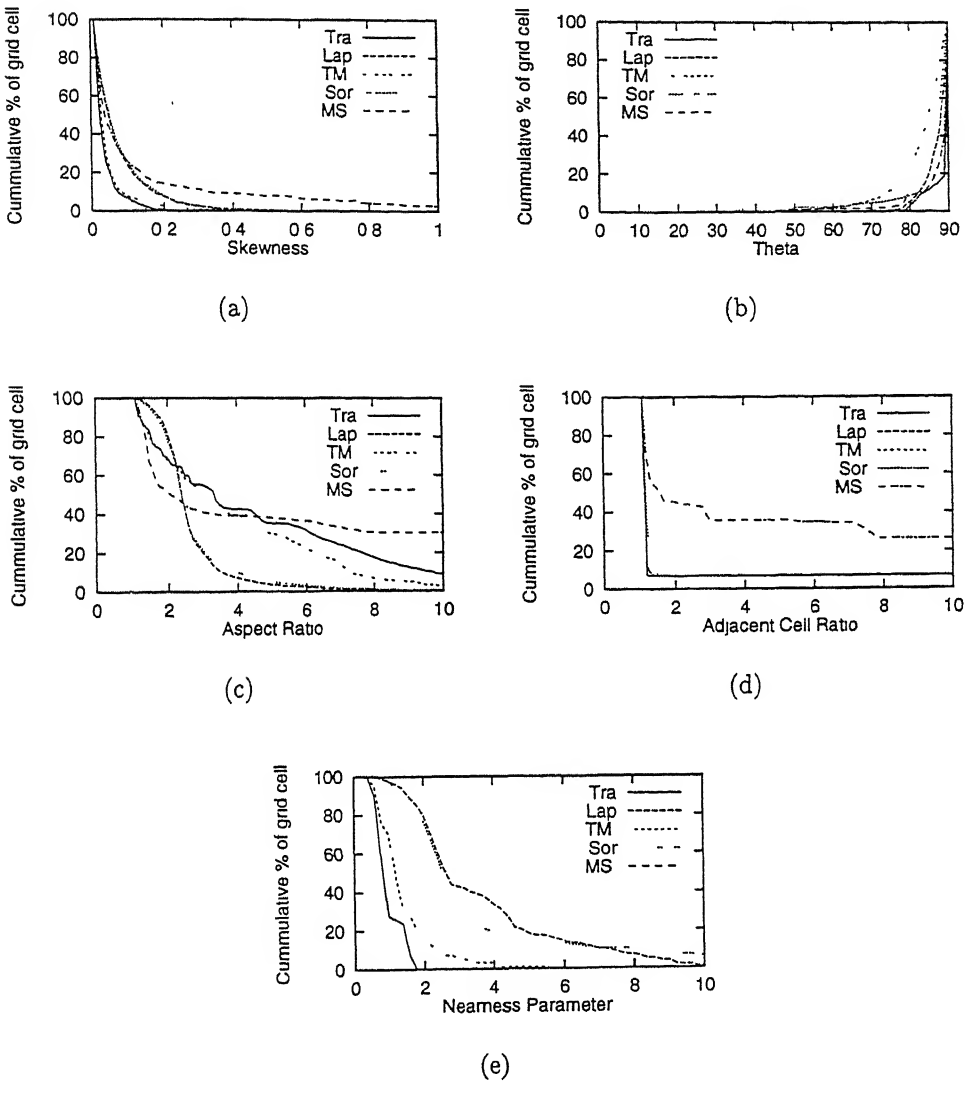
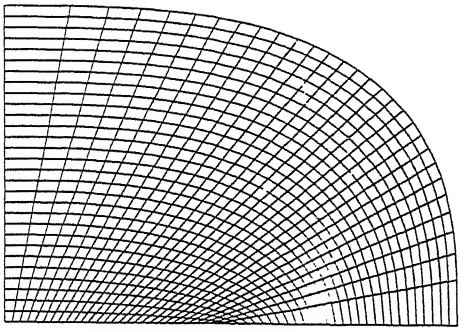
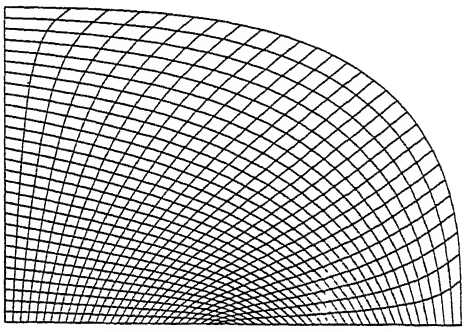


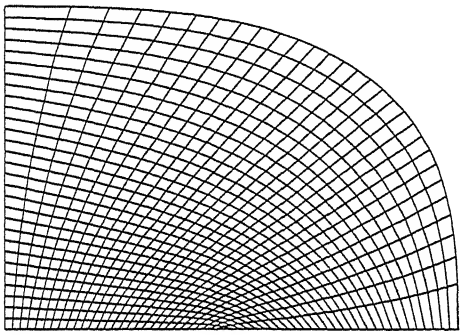
Figure 4.16: Parameters Plot for Rectangular Channel Section with a Slope with Nonuniform Boundary Point Distribution



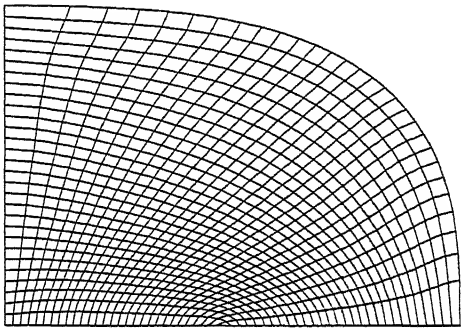
(a) Transfinite interpolation



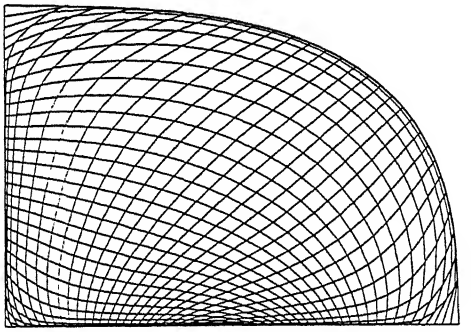
(b) Laplace method



(c) Thomas-Middlecoff Method



(d) Sorenson Method



(e) Modified Sorenson Method

Figure 4.17: Quarter Portion of a Super Ellipse with Uniform Boundary Point Distribution

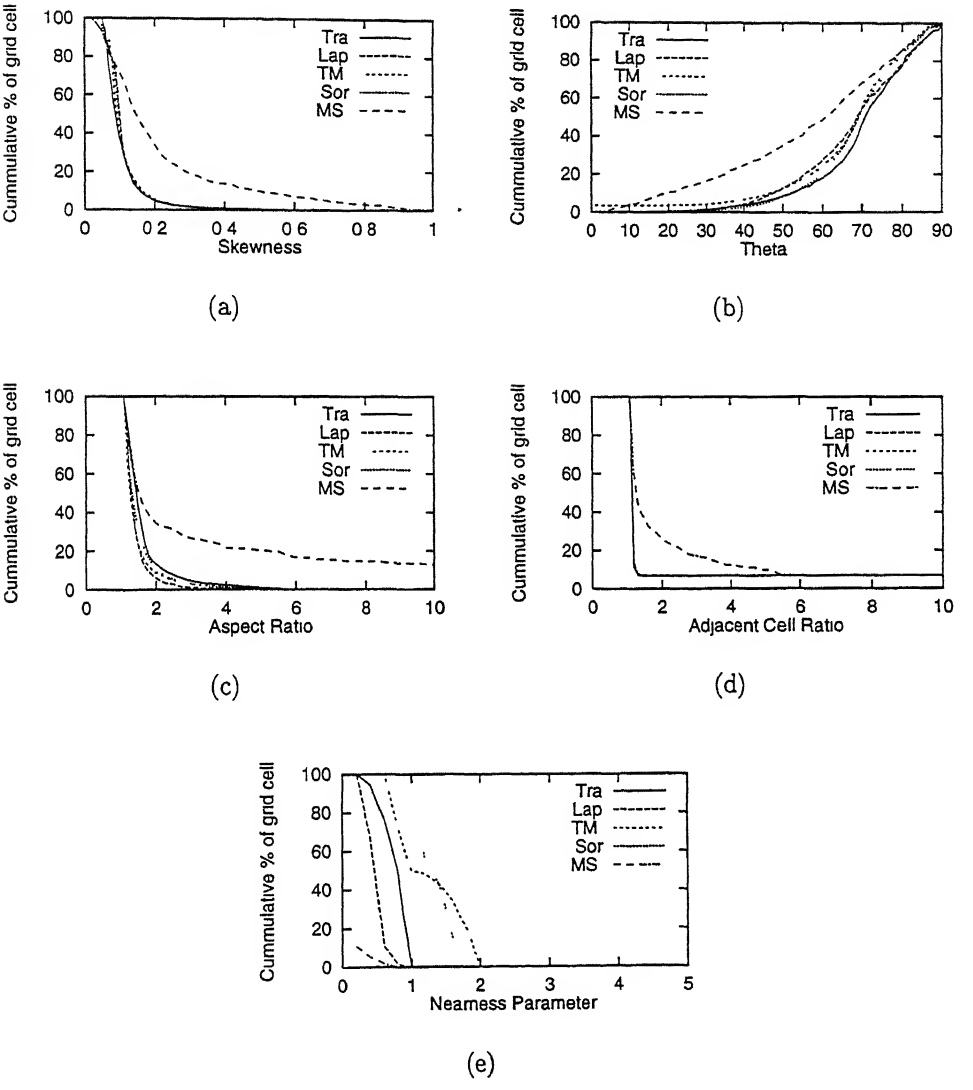
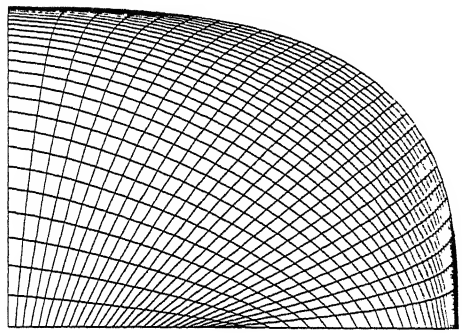
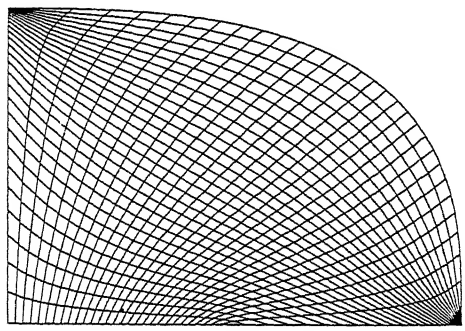


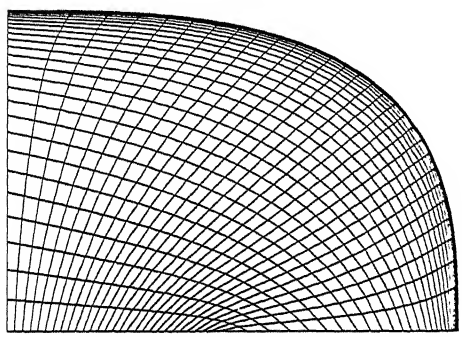
Figure 4.18: Parameters Plot for Quarter Portion of a Super Ellipse with Uniform Boundary Point Distribution



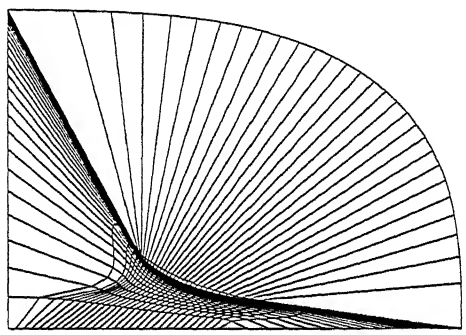
(a) Transfinite interpolation



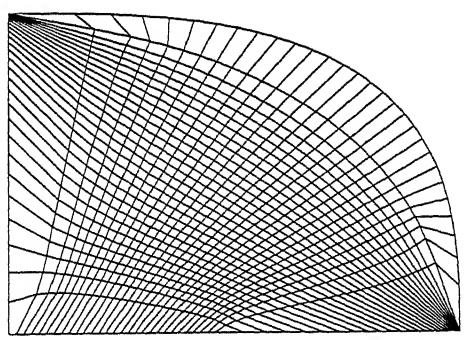
(b) Laplace method



(c) Thomas-Middlecoff Method



(d) Sorenson Method



(e) Modified Sorenson Method

Figure 4.19: Quarter Portion of a Super Ellipse with Nonuniform Boundary Point Distribution

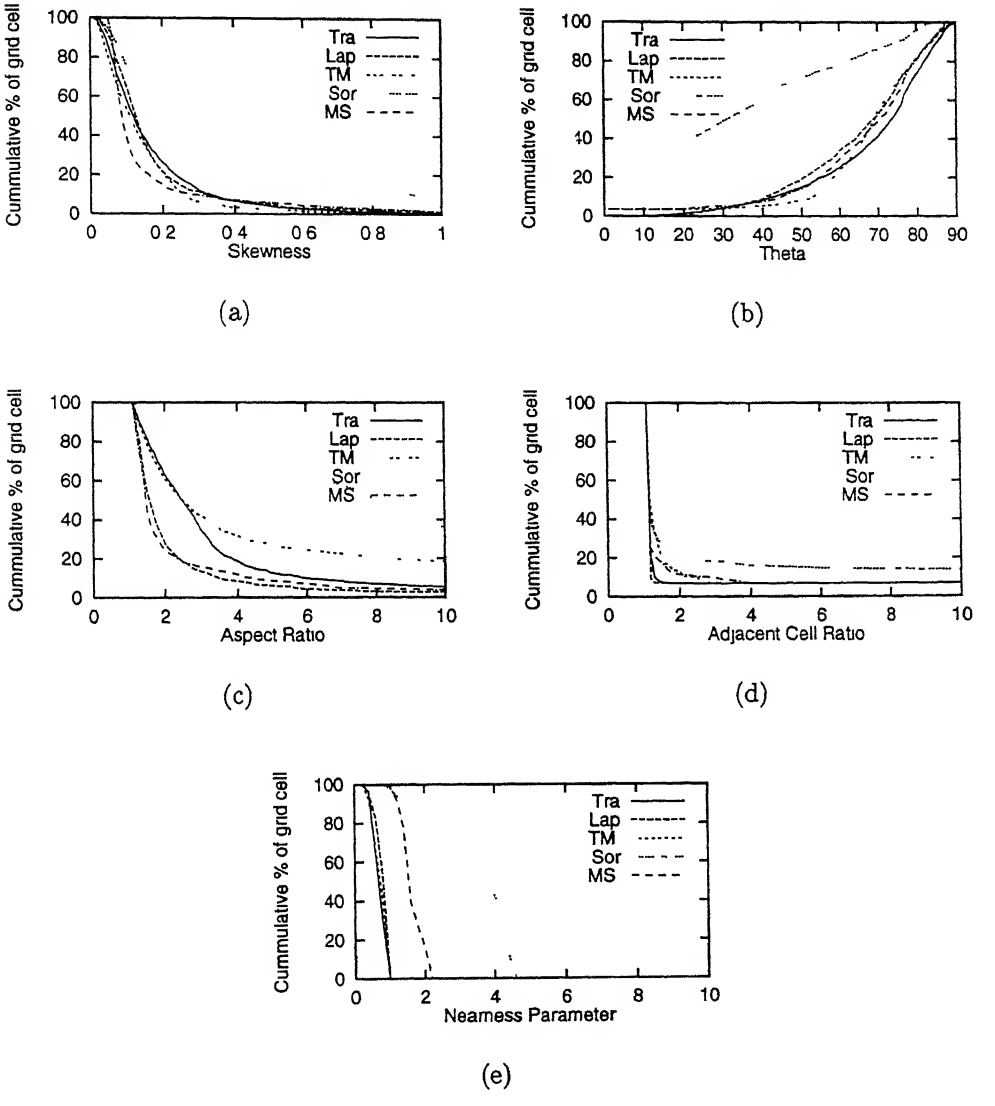


Figure 4.20: Parameters Plot for Quarter Portion of a Super Ellipse with Nonuniform Boundary Point Distribution

Bibliography

[1] Anderson, D.A. and Steinbrenner J., "Generating Adaptive Grids with a Conventional Grid Scheme", *AIAA Paper 86-0427*, 1986.

[2] Anderson D.A. Tannehill J.C. and Pletcher R.H., "*Computational Fluid Mechanics and Heat Transfer*", Hemisphere, Newyork, 1984.

[3] Eiseman P.R., "Automatic Algebraic Grid Generation", In J.F. Thompson, editor, *Numerical Grid Generation*, 1982.

[4] Gordon W.J. and Thiel L.C., "Transfinite Mapping and Their Applications to Grid Generation", In J.F. Thompson, editor, *Numerical Grid Generation*, 1982.

[5] Mastin C.W , "Error Induced by Coordinate Systems", In J.F. Thompson, editor, *Numerical Grid Generation*, 1982.

[6] Smith R.E., "Algebraic Grid Generation", In Thompson J.F., editor, *Numerical Grid Generation*, 1982.

[7] Sorenson R.L., "A Computer Program to Generate Two Dimensional Grids about Airfoils and Other Shapes by the Use of Poissons Equation", Technical Report TM–81198, NASA, NASA Ames Research Centre, 1980.

[8] Steger A.J. and Sorenson R.L., "Automatic Mesh Point Clustering near a Boundary in Grid Generation with Elliptic Partial Differential Equations", *Journal of Computational Physics*, 33(3):405–410, 1979.

[9] Sundararajan T., "Automatic Grid Generation for Complex Geometry Problems", In Muralidhar K. and Sundararajan T., editors, *Computational Fluid Flow and Heat Transfer*, pages 459 – 509, Newdelhi, 1995. Narosa.

[10] Thomas P.D., "Numerical Generation of Composite Three Dimensional Grids by Quasilinear Elliptic Systems", In Thompson J.F., editor, *Numerical Grid Generation*, 1982.

[11] Thomas P.D. and Middlecoff J.F., "Direct Control of the Grid Point Distribution in Meshes Generated by Elliptic Equations", *AIAA J*, 18(6):652–656, 1980.

[12] Thompson J.F., "TOMCAT – A Code for Numerical Generation of Boundary - Fitted Curvilinear Coordinate Systems on Fields Containing any Number of Arbitary Two Dimensional Bodies", *Journal of Computational Physics*, 24:245 – 273, 1977.

[13] Thompson J.F., "Elliptic Grid Generation", In Thompson J.F., editor, *Numerical Grid Generation*, pages 79 – 105, 1982.

[14] Thompson J.F., editor. "*Numerical Grid Generation*", North – Holland, 1982.

[15] Thompson J.F Thames F.C. and Mastin C.W., "Automatic Numerical Generation of Body-Fitted Curvilinear Coordinate System for Field Containing any Number of Arbitrary Two-Dimensional Bodies", *Journal of Computational Physics*, 15:299 – 319, 1974.

[16] Thompson J.F. Warsi Z.U.A. and Mastin C.W., "*Numerical Grid Generation – Foundations and Applications*", North–Holland, 1985.